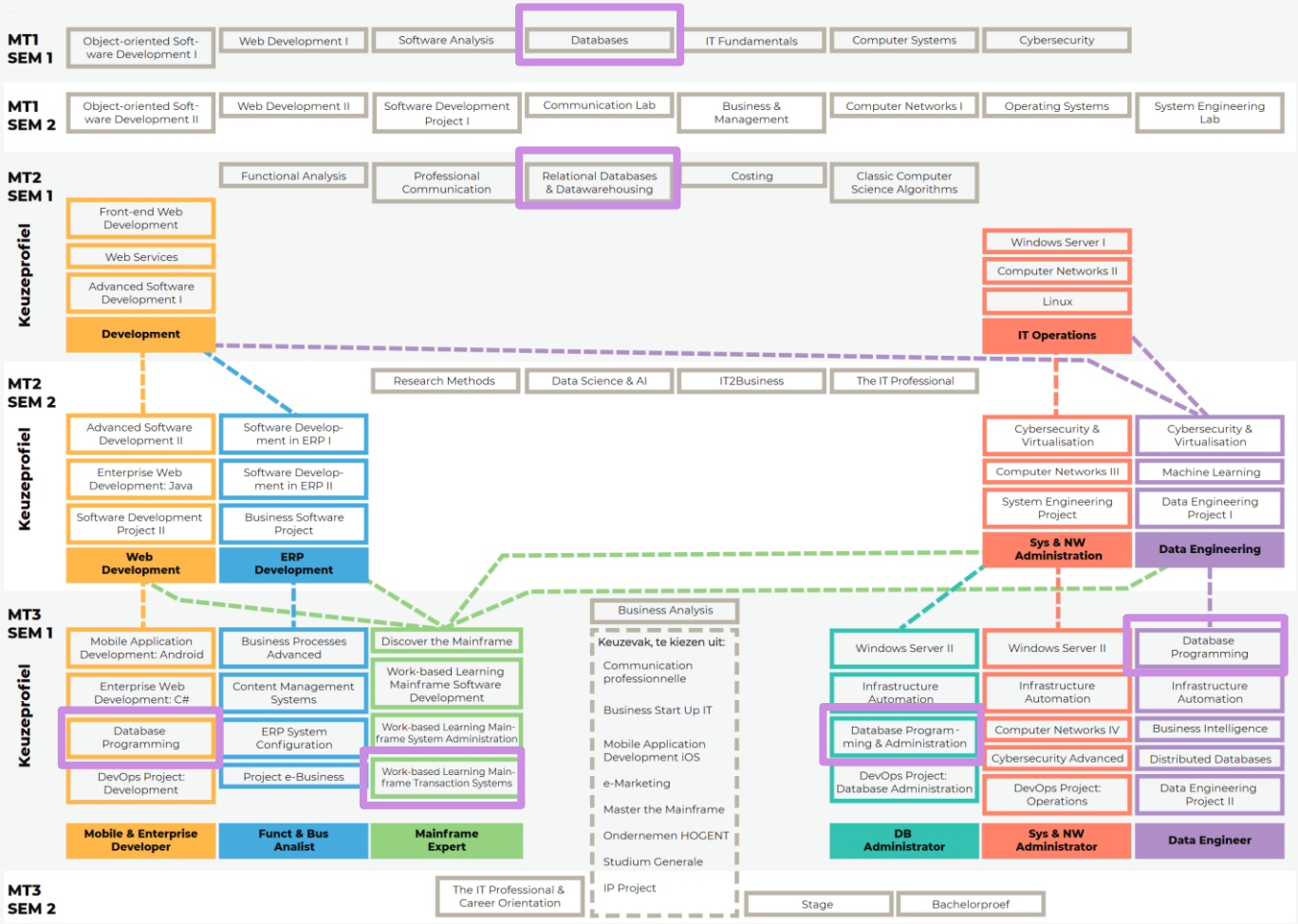
Databases

HO
GENT

Situering

HO GENT


MT1 SEM 1
Object-oriented Software Development I
Web Development I
Software Analysis
Databases
IT Fundamentals
Computer Systems
Cybersecurity
MT1 SEM 2
Object-oriented Software Development II
Web Development II
Software Development Project I
Communication Lab
Business & Management
Computer Networks I
Operating Systems
System Engineering Lab
MT2 SEM 1
Functional Analysis
Professional Communication
Relational Databases & Datawarehousing
Costing
Classic Computer Science Algorithms
Keuzeprofiel
Front-end Web Development
Web Services
Advanced Software Development I
Development
Windows Server I
Computer Networks II
Linux
IT Operations
MT2 SEM 2
Research Methods
Data Science & AI
IT2Business
The IT Professional
Keuzeprofiel
Advanced Software Development II
Enterprise Web Development: Java
Software Development Project II
Web Development
Software Development in ERP I
Software Development in ERP II
Business Software Project
ERP Development
Cybersecurity & Virtualisation
Computer Networks III
System Engineering Project
Sys & NW Administration
Cybersecurity & Virtualisation
Machine Learning
Data Engineering Project I
Data Engineering
MT3 SEM 1
Keuzeprofiel
Mobile Application Development: Android
Enterprise Web Development: C#
Database Programming
DevOps Project: Development
Mobile & Enterprise Developer
Business Processes Advanced
Content Management Systems
ERP System Configuration
Project e-Business
Funct & Bus Analist
Discover the Mainframe
Work-based Learning Mainframe Software Development
Work-based Learning Mainframe System Administration
Work-based Learning Mainframe Transaction Systems
Mainframe Expert
Business Analysis
Keuzevak, te kiezen uit:
Communication professionnelle
Business Start Up IT
Mobile Application Development iOS
e-Marketing
Master the Mainframe
Ondernemen HOGENT
Studium Generale
IP Project
Windows Server II
Infrastructure Automation
Database Programming & Administration
DevOps Project: Database Administration
DB Administrator
Windows Server II
Infrastructure Automation
Computer Networks IV
Cybersecurity Advanced
DevOps Project: Operations
Sys & NW Administrator
Database Programming
Infrastructure Automation
Business Intelligence
Distributed Databases
Data Engineering Project II
Data Engineer
MT3 SEM 2
The IT Professional & Career Orientation
Stage
Bachelorproef

HO GENT

Store and Manage

De student kan complexe verzamelde informatie analyseren en verwerken tot de meest geschikte gestructureerde databases die hij/zij - met het oog op performantie - efficiënt beheert en gebruikt.

HO GENT

Doelstellingen-ECTS

- Kan de verschillende types databanken toelichten en kan die situeren tegenover de “klassieke” bestanden.
- Kan een (E)ERD opstellen door relevante entiteitstypes, attribuuttypes, de relaties tussen de entiteitstypes en de cardinaliteiten van de relaties af te leiden uit ongestructureerde informatiebron(nen).
- Kan een bestaand (E)ERD evalueren op juistheid en aanpassen op basis van bijkomende of gewijzigde informatie.
- Kan een conceptueel model ((E)ERD) omzetten naar een relationeel datamodel.
- Kan een relationeel datamodel maken met behulp van de normalisatietechniek.
- Kan een relationeel datamodel evalueren op 1NV, 2NV of 3NV.
- Kan eenvoudige queries uitvoeren via SQL
  (SELECT – FROM – WHERE – GROUP BY – HAVING – JOIN).
- Kan de data in een databank bijwerken via SQL
  (INSERT – UPDATE – DELETE).
- Kan de structuur van een databank bijwerken via SQL
  (CREATE-ALTER-DROP).
- Kan de definities i.v.m. verzamelingen, relaties toepassen.

HO GENT

Leerinhoud

- H1: Inleiding tot Databanken
- H2-H4: Omzetten van tekst of formulier naar een conceptueel model ((E)ERD)
- H5: Omzetten van een conceptueel model naar een relationeel model
- H6: Normalisatie
- H7-10: SQL
- H11: Verzamelingenleer

HO
GENT

Lesgevers

Gent:

- Gertjan Bosteels
- Sabine De Vreese
- Thomas Parmentier
- Olivier Rosseel
- Sharon Van Hove

Aalst:

- Joeri Van Herreweghe

Virtuele campus + Afstandsonderwijs:

- Sabine De Vreese

HO GENT

Cursusmateriaal

- Cursus: Chamilo
- Presentaties: Chamilo
- Workshops: Github

Tools

- DrawIO
- MySQL

HO GENT

Evaluatie

Eerste examenkans:

- Permanente evaluatie: 15 %
  - Test EERD (1 uur):
    - » Week 7 (week van 6 november 2023)
    - » Concrete info: zie mededeling op Chamilo
- Examen: 85%
  - Deel datamodellering: 55 %
  - Deel SQL: 30 %

Tweede examenkans:

- Examen: 100%
  - Deel datamodellering: 70 %
  - Deel SQL: 30 %

HO GENT

Databases

H1 Databanken gekaderd

HO
GENT

H1. Databanken gekaderd

1. Wie gebruikt databanktechnologie
2. Basisbegrippen
3. Gegevensbeheer
4. Delen van een databanksysteem
5. Kenmerken van databanken en databankbeheer

HO GENT

1. Wie gebruikt databanktechnologie

HO
GENT

Wie gebruikt databank(technologie)?

- Traditionele bedrijfsapplicaties (loonberekening, tijdsregistratie, …)
- Biometrische applicaties (vingerafdrukken, resultaten scans)
- Sensor-applicaties (in kerncentrales, …)
- GIS applicaties (geografische informatie systemen (Google Maps, …)
- Big Data applicaties (Walmart, …)
- ‘Internet of Things (IoT)’ applicaties (Telematics, …)

- → opslag en terug ophalen van informatie (data)

HO GENT

2. Basisbegrippen

HO
GENT

Basisdefinities: databank.

Een gedeelde verzameling van logisch met elkaar verbonden gegevens en hun beschrijving, ontworpen om aan de informatienoden van een organisatie te voldoen.
(T. Connolly).

- digitaal opgeslagen
- specifiek bedrijfsproces
- specifieke groep (gebruikers en applicaties)

HO GENT

Basisdefinities: DBMS.

- Een Database Management System (DBMS):
  - een **verzameling** computerprogramma's (softwaremodules)
  - Nodig om een databank te
    - definiëren
    - creëren
    - wijzigen
    - beheren
    - gebruiken (gegevens invoeren en 'lezen').

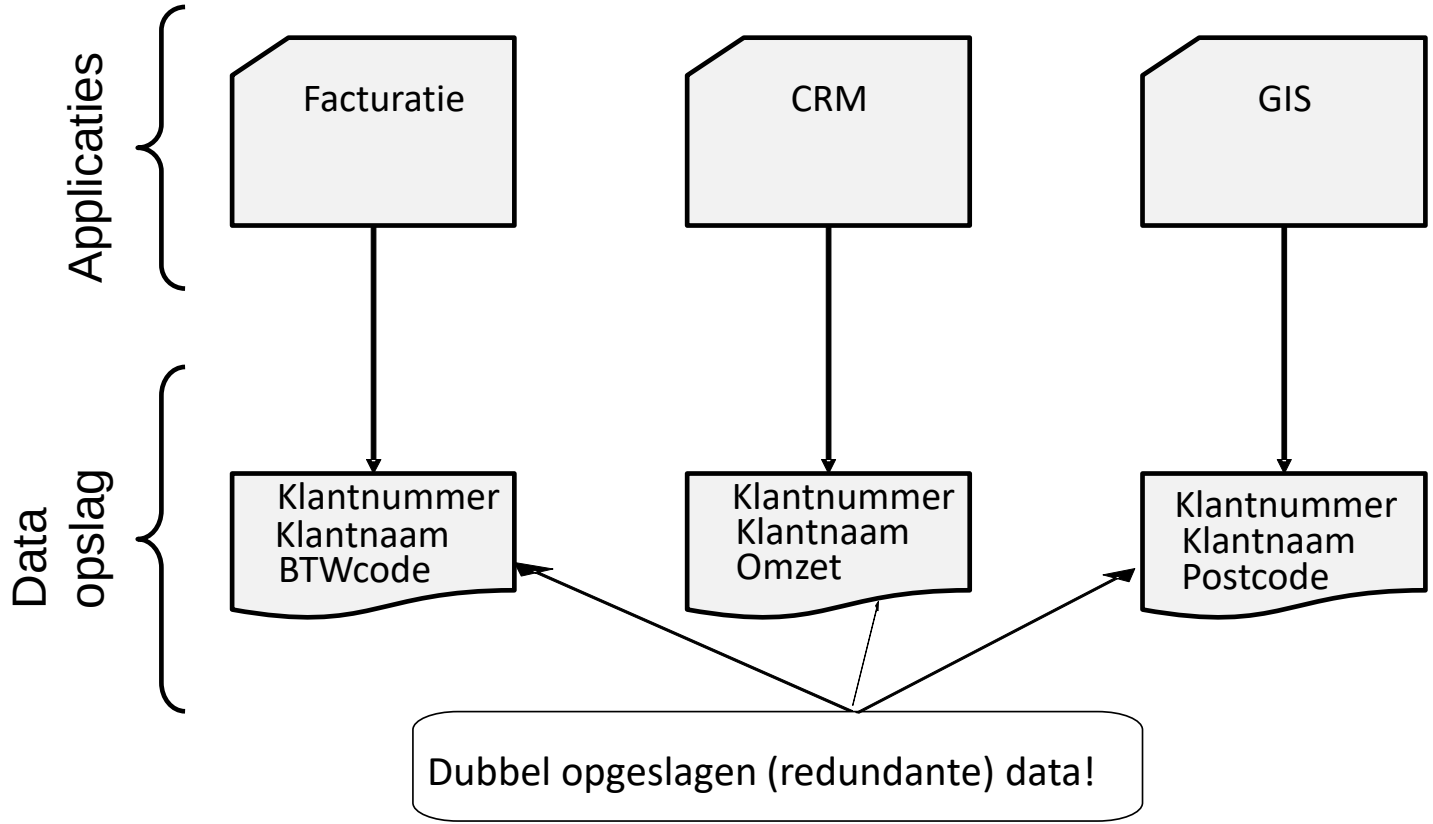
- Databanksysteem: databank + DBMS

HO GENT

3. Gegevensbeheer

HO
GENT

Gedecentraliseerd <-> gecentraliseerd.



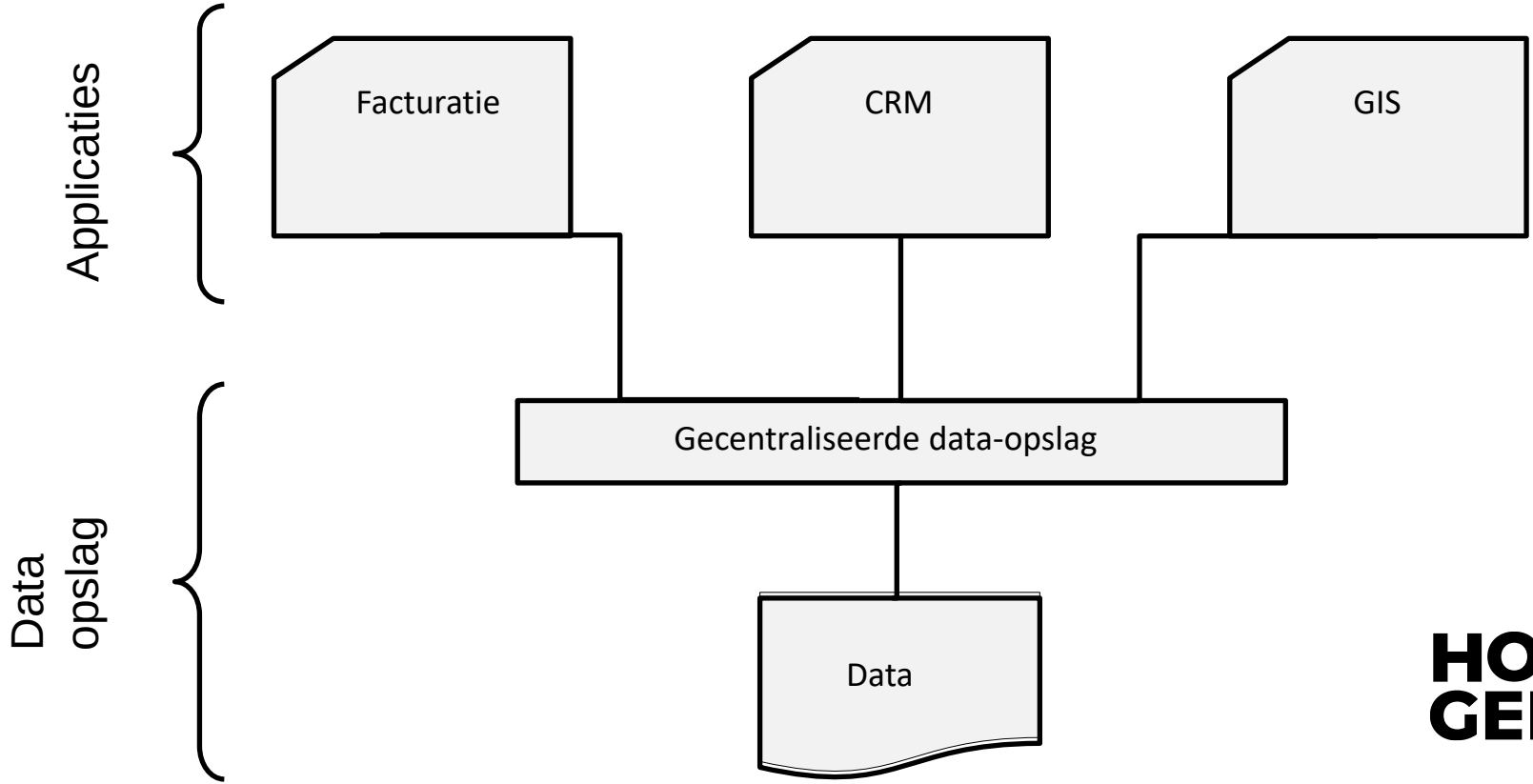
HO GENT

Data management.

- Gedecentraliseerde aanpak:
  - Dubbele of redundante opslag
  - Risico op inconsistente data
  - Sterke koppeling tussen applicaties en data
  - Concurrente toegang moeilijk te realiseren
  - Applicaties voor meerdere diensten/bedrijven moeilijk te realiseren

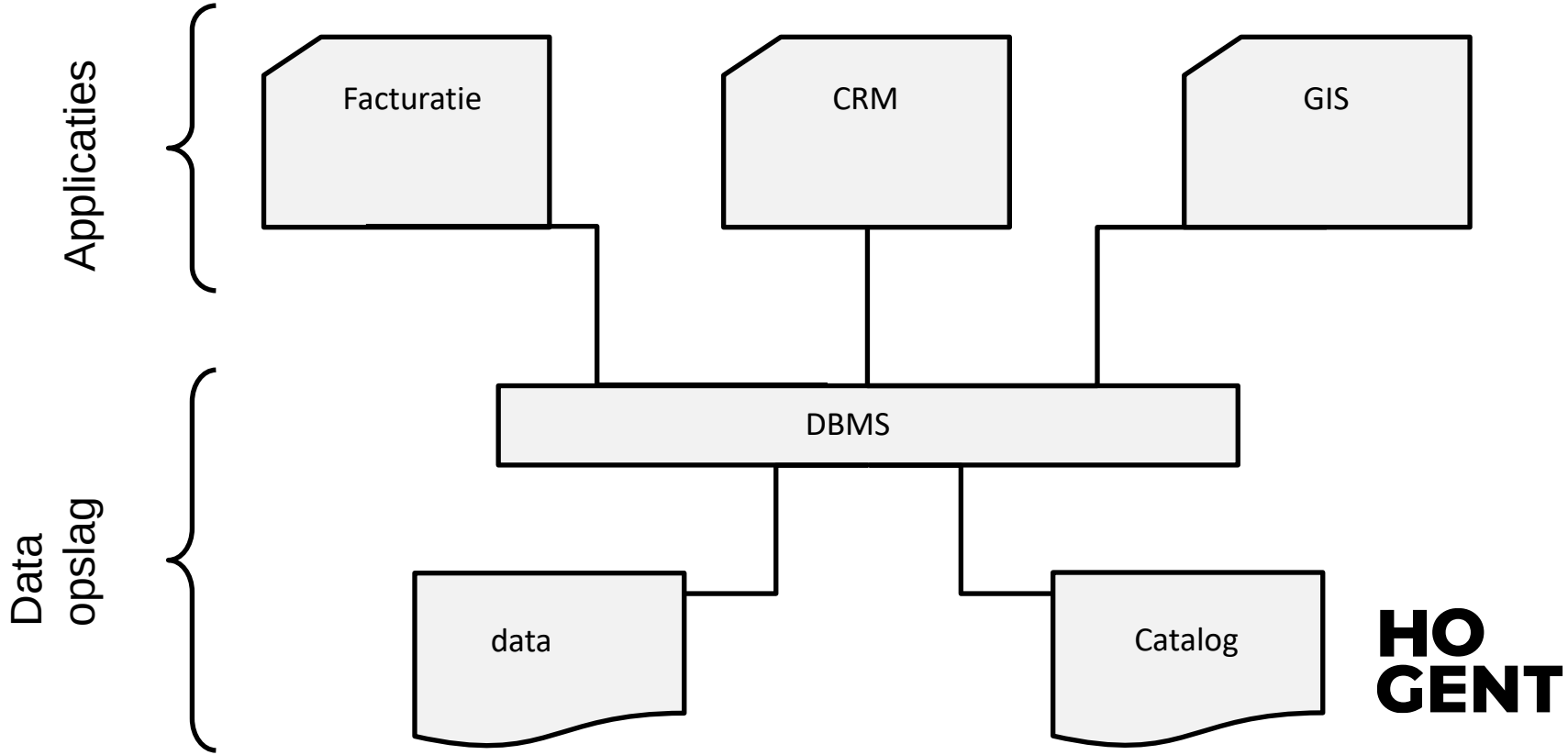
HO GENT

Data management.



HO GENT

Data management.



HO GENT

Data management.

- Gecentraliseerde aanpak:
  - Efficiënter
  - Consistente data
  - Eenvoudiger te beheren
  - Losse koppeling tussen applicaties en data
  - DBMS biedt mogelijkheden bij het concurrent ‘beheren’ van data

HO GENT

Data management.

- (Txt) Bestand

```
Procedure FindCustomer;
begin
 open file Customer.txt;
Read(Customer)
While not EOF(Customer)
If Customer.name='Bart' then
display(Customer);
EndIf
Read(Customer);
EndWhile;
End;
```

- (Relationele) Databank (SQL)

```
SELECT *
FROM Customer
WHERE name = 'Bart'
```

HO GENT

4. Delen van een databanksysteem

HO
GENT

Elementen van een databanksysteem.

- Databankmodel versus instances
- Data model
- 3-lagen architectuur
- Catalog
- Databankgebruikers

HO
GENT

Databankmodel versus instances.

- Databankmodel = databankschema
  - bevat
    - beschrijving van de databankstructuur
    - specificaties v/d elementen, hun eigenschappen, relaties, beperkingen, …
  - Opgesteld tijdens databankontwerp
  - Wijzigt niet om de haverklap
  - Opgeslagen in de catalog
- Toestand van een databank
  - Op dat ogenblik aanwezige data
  - Wijzigt voortdurend

HO GENT

Databankmodel versus instances.

- Voorbeeld databankmodel :

  Kunstenaar (naam, geboorteplaats, geboortedatum)
  Kunstwerk (naam, museum, jaar)
  Museum (naam, stad)

HO GENT

Databankmodel versus instances.

- Toestand van een databank :

MUSEUM

naam	stad
Sint-Pietersbasiliek	Rome
Museo Reina Sofia	Madrid
Gemäldegalerie van de Staatliche Museen	Berlijn

KUNSTENAAR

naam	geboorteplaats	geboortedatum
Michelangelo	Caprese	06/03/1475
Rembrandt	Leiden	15/07/1606
Picasso	Malaga	25/08/1881

KUNSTWERK

naam	museum	jaar
Guernica	Museo Reina Sofia	1937
Christuskop	Gemäldegalerie van de Staatliche Museen	1648
Pieta	Sint-Pietersbasiliek	1499

HO GENT

Datamodel.

- Gegevensmodel: weergave van de gegevens met hun kenmerken en hun relaties

- Conceptueel gegevensmodel: perfecte weergave van de gegevensvereisten van de ‘business’
  - opgesteld met de business
  - omgezet naar logisch en intern model

HO GENT

Gegevensmodel.

- Conceptueel gegevensmodel :
  - algemene beschrijving gegevenselementen, kenmerken en relaties
    - Gebruikt door ‘IT’ en ‘business’
    - Weergave hoe ‘de business’ de gegevens ziet
    - Voorstelling: (E)ERD diagram

- !!Veronderstellingen en ontbrekende informatie duidelijk vermelden!!

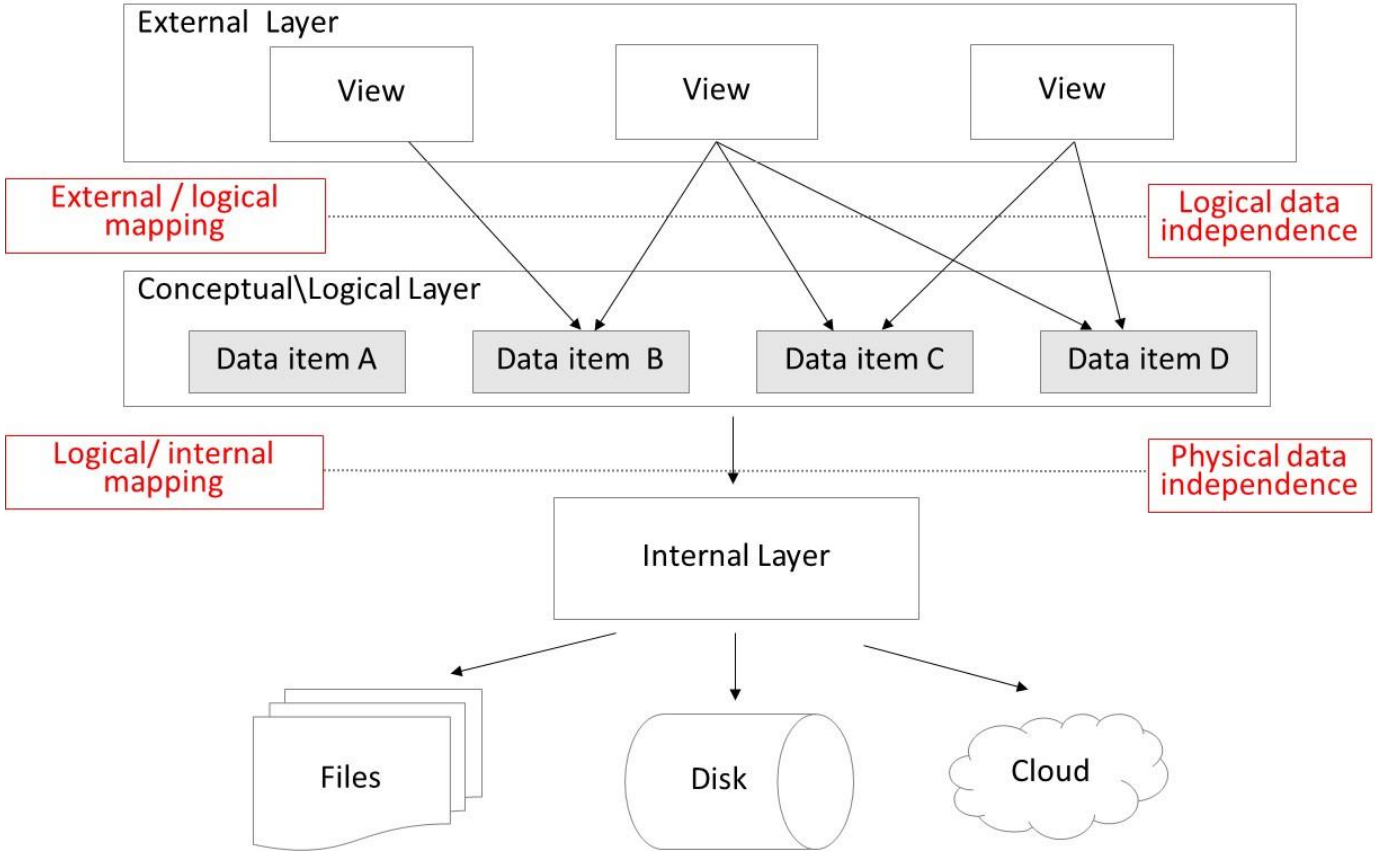
HO GENT

Gegevensmodel.

- Logisch gegevensmodel
  - vertaling conceptueel gegevensmodel naar het type databankmodel
    - Relationeel, hiërarchisch, OO, XML, NoSQL
  - omzetten naar intern (fysiek) gegevensmodel
    - Geeft informatie over fysieke opslag:
      - waar worden welke gegevens opgeslagen
      - Onder welke vorm
      - Indexen die het ophalen versnellen
    - Zeer DBMS afhankelijk
  - Externe gegevensmodellen
    - Deelverzameling van het logisch model
    - Voor een specifieke doelgroep.

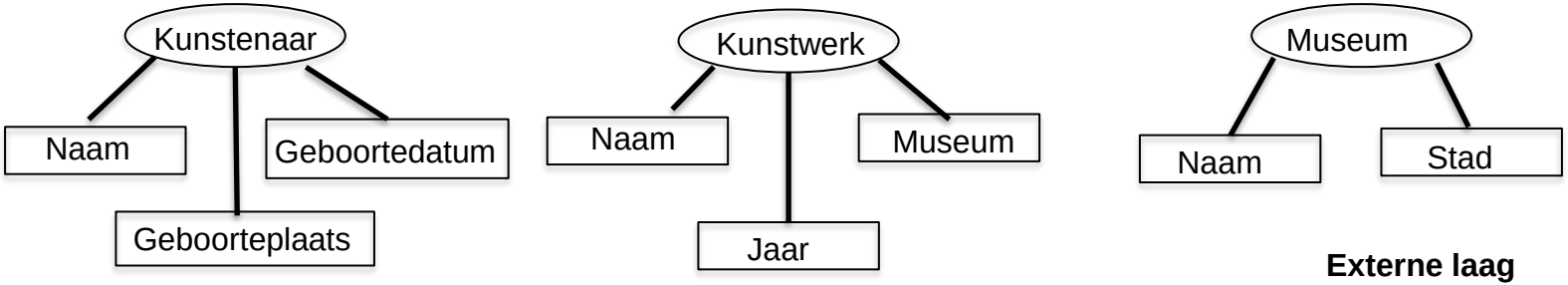
HO GENT

3-Lagen model.



HO GENT

3-Lagen model.



Externe laag

Kunstenaar (naam, geboorteplaats, geboortedatum)
Kunstwerk (naam, museum, jaar)
Museum (naam, stad)

Logische laag

Interne laag

HO GENT

Catalog.

- Schatkist van DBMS
- Definities en beschrijving van de elementen in de DB (= metadata)
- Definities logisch gegevensmodel en intern gegevensmodel
- Zorgt voor synchronisatie en consistentie van de gegevensmodellen

HO GENT

Databank gebruikers.

- IT Architect ontwerpt conceptueel gegevensmodel in samenspraak met de ‘business’
- Dbontwerper vertaalt conceptueel model naar logisch en intern model
- DBA (databankbeheerder of database administrator) implementeert en monitort DB
- Applicatieontwikkelaar schrijft databankprogramma’s/databankapplicaties
- ‘Business’ gebruikt databankapplicaties en voert op die manier databankacties uit

HO GENT

5. Kenmerken van databanken en databankbeheer

HO
GENT

Nog enkele belangrijke begrippen.

- Gegevensonafhankelijkheid
- Modellen
- Gestructureerde en ongestructureerde gegevens
- Integriteitsregels
- Redundante gegevens

HO GENT

Gegevensonafhankelijkheid.

- = wijzigingen aan de gegevensbeschrijving hebben weinig tot geen impact op de applicaties

- Fysieke gegevensonafhankelijkheid: wijzigingen van de opslagspecificaties hebben geen invloed op het logisch model noch op de applicatie
  → wordt opgevangen door het DBMS

- Logische gegevensonafhankelijkheid: minimale aanpassingen aan de applicaties bij wijzigingen aan het logisch model

HO GENT

(On)gestructureerde gegevens.

- Gestructureerde gegevens
  - Kunnen in een logisch datamodel voorgesteld worden
  - Integriteitsregels kunnen opgesteld en afgedwongen worden
  - Vereenvoudigen, opzoeken, verwerken en analyseren
  - Voorbeelden: naam, geboortejaar, geboorteplaats van een kunstenaar
- Ongestructureerde gegevens
  - kunnen niet op een zinvolle manier worden geïnterpreteerd door een applicatie
  - Voorbeelden: gesprekken op social media, e-mails
  - Let op: er bestaat veel meer ongestructureerde data dan gestructureerde data

HO GENT

(On)gestructureerde gegevens.

- Semi-gestructureerde gegevens
  - De structuur van de gegevens is zeer onregelmatig of zeer wisselend.
  - Voorbeelden: webpagina's van individuele gebruikers op een social media platform, samenvattende human resources documenten

HO GENT

Integriteitsregels.

- Integriteitsregels worden gedefinieerd op basis van het conceptueel model en opgeslagen in de catalog
  - Worden afgedwongen door het DBMS
- Vastleggen hoe gegevens worden opgeslagen (syntactische regel)
  - Voorbeeld: geboortedatum, eenheidsprijs
- Vastleggen wanneer gegevens correct zijn (semantische regel)
  - Voorbeeld: Eenheidsprijs > 0;
    geboortedatum niet > vandaag

HO GENT

Redundante gegevens.

- Databank = centrale en unieke opslag gegevens

- Soms worden databanken gedupliceerd uit veiligheidsoverwegingen of omwille van performantie →redundantie

- DBMS is verantwoordelijk voor de synchronisatie en garandeert de juistheid van de gegevens

HO GENT

Indeling van DBMS-systemen.

- Op basis van het gegevensmodel
- Op basis van het gebruik

HO
GENT

Indeling op basis van het gegevensmodel.

- Hiërarchisch DBMS
  - Het gegevensmodel is een omgekeerde boom
  - DML is procedureel en gebaseerd op ‘recordverwerking’
  - Geen query processor (logisch en fysisch datamodel lopen door elkaar)
  - Voorbeeld: IMS (IBM)

- Netwerk DBMS
  - Gebruiken een netwerk gegevensmodel
  - CODASYL DBMS
  - DML is procedureel en gebaseerd op ‘recordverwerking’
  - geen query processor (logisch en fysisch datamodel lopen door elkaar)
  - Voorbeeld: CA-IDMS (Computer Associates → Broadcom)

HO GENT

Indeling op basis van het gegevensmodel.

- Relationeel DBMS
  - Maakt gebruik van het relationeel gegevensmodel
  - Momenteel het meest frequent gebruikt in de bedrijfswereld
  - SQL (beschrijvend en gebaseerd op resultset)
  - Query processor
  - Strikte scheiding tussen het logisch en het fysisch gegevensmodel
  - Voorbeelden: MySQL (open source, Oracle), Db2 (IBM), Oracle DBMS (Oracle), SQLServer (Microsoft)

HO GENT

Indeling op basis van het gegevensmodel.

- Object-Oriented DBMS (OODBMS)
  - Gebaseerd op het OO gegevensmodel
  - Geen probleem om het gegevensmodel van de databank te verenigen met de OO programmeertaal
  - Voorbeelden: db4o (open source, Versant), Caché (Intersystems), GemStone/S (GemTalk Systems)
  - Alleen doorgebroken in niche markten omwille van de complexiteit van het OODBMS

HO GENT

Indeling op basis van het gegevensmodel.

- Object-Relationeel DBMS (ORDBMS)
  - Wordt ook extended relationeel DBMS (ERDBMS) genoemd
  - Gebruikt een relationeel gegevensmodel uitgebreid met OO concepten
  - DML is SQL (beschrijvend en gebaseerd op resultset)
  - Voorbeelden: Oracle DBMS (Oracle), DB2 (IBM), SQLServer (Microsoft)

HO GENT

Indeling op basis van het gegevensmodel.

- XML DBMS
  - Maakt gebruik van het XML gegevensmodel om gegevens op te slaan
  - Native XML DBMS (voorbeelden: BaseX, eXist) zal de boomstructuur van een XML document projecteren op een fysische opslagstructuur
  - XML-enabled DBMS (voorbeelden: Oracle, IBM Db2) zijn bestaande DBMS-systemen die uitgebreid zijn met faciliteiten om gegevens uit een XML gegevensmodel op te slaan

HO GENT

Indeling op basis van het gegevensmodel.

- NoSQL DBMS
  - Bedoeld om massaal veel en ongestructureerde data op te slaan
  - Kan ingedeeld worden in
    - key-value stores (sleutelwaardedatabanken)
    - column-oriented databases (gegevens worden per kolom gestockeerd ipv per rij)
    - graph databases
  - Focust op schaalbaarheid van de databank en op werken met onregelmatige en snel veranderende gegevensstructuren
  - Voorbeelden: Apache Hadoop, MongoDB, Neo4j

HO GENT

Indeling op basis van gebruik.

- On-line transaction processing (OLTP)
  - Focust op het beheren en verwerken van operationele en transactionele gegevens
  - De databankserver moet heel veel eenvoudige transacties per tijdseenheid kunnen verwerken
  - Het DBMS moet goed ontworpen zijn om heel veel korte eenvoudige queries uit te voeren
- On-line analytical processing (OLAP)
  - Focust op het gebruiken van operationele gegevens om strategische en tactische beslissingen te nemen
  - Een beperkt aantal gebruikers zal complexe queries uitvoeren
  - Het DBMS moet complexe queries efficiënt kunnen verwerken.

HO GENT

Indeling op basis van gebruik.

- Big Data & Analytics
  - NoSQL databanken
  - Focust op meer flexibele databankstructuren en databanken zonder schema
  - Stockeert ongestructureerde data zoals e-mails, Twitter tweets, Facebook posts, ...
- Multimedia
  - Multimedia DBMS-systemen zorgen voor opslagruimte voor multimedia data zoals tekst, foto's, audio, video, 3D games, ...
  - Moet ook de bijhorende query mogelijkheden (op basis van inhoud) aanbieden

HO GENT

Indeling op basis van gebruik.

- Geometrische toepassingen
  - Een geometrisch DBMS laat toe om geometrische gegevens op te slaan en op te vragen (zowel 2D als 3D)

    Voorbeeld: Geografische Informatie Systemen (GIS)

- Sensoren
  - Een sensor DBMS beheert de gegevens van sensoren
    Voorbeelden: biomedische gegevens van kledij of medische apparatuur of gegevens uit telematica toepassingen

HO GENT

Indeling op basis van gebruik.

- Mobiel
  - Een mobiele DBMS werkt op een mobiel toestel (smartphone, tablet , …)
  - Moet altijd online zijn en kunnen werken in een omgeving met een beperkte verwerkingskracht, weinig opslagruimte en beperkte energievoorziening (batterij)
- Open source
  - De code van een open source DBMS is voor iedereen toegankelijk en iedereen kan code toevoegen
  - Zie ook: www.sourceforge.net
  - Voorbeeld: MySQL (Oracle)

HO GENT

Databases

H2 conceptueel model

HO
GENT

H2. Conceptueel model

1. Fasen in databankontwerp
2. Entity Relationship Diagram
3. Oefeningen
4. Bronnen

HO
GENT

1. Fasen in databankontwerp

HO GENT

Fasen in databankontwerp

Het ontwerpen van een databank vertrekt vanuit de bedrijfsprocessen en bestaat uit 4 fases:

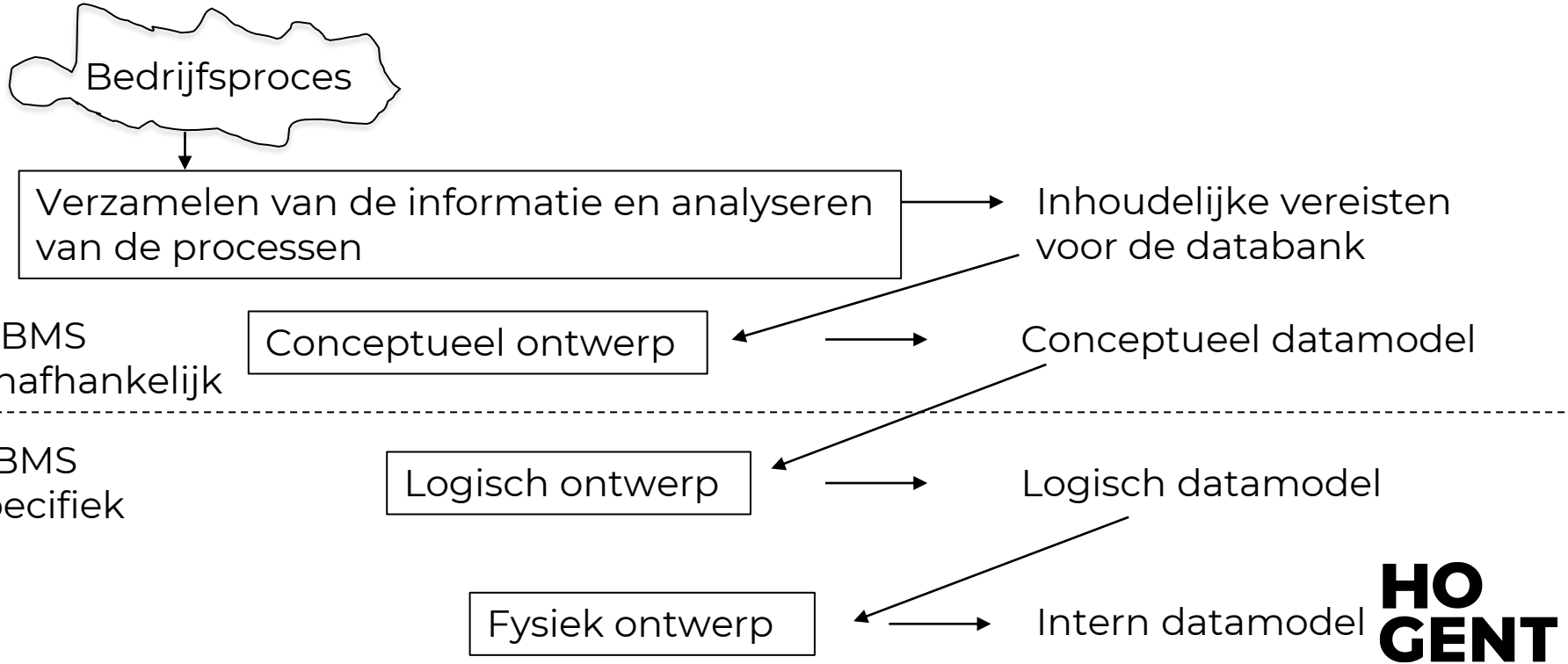
- Fase 1 = Verzamelen en analyseren van de functionele / inhoudelijke vereisten
- Fase 2 = Conceptueel ontwerp
- Fase 3 = Logisch ontwerp
- Fase 4 = Fysiek ontwerp

Mogelijke bedrijfsprocessen:

- het maken van facturen
- werkroosters en prestaties van werknemers
- voorraadbeheer
- puntenadministratie
- …

HO GENT

Fasen in databankontwerp



HO GENT

Fase 1 = Verzamelen en analyseren informatie

- Doel: de stappen en de benodigde data van het bedrijfsproces begrijpen.
  Wat nemen we op in de databank?

- Dit kan via
  - interviews met de opdrachtgever
  - analyse van bestaande formulieren en rapporten

- Vragen die moeten beantwoord worden
  - Welke data moet in de databank worden opgenomen?
  - Wat is de betekenis en context van alle data, symbolen, gebruikte coderingen?
  - Hoe zal de data worden verwerkt?
  - Wat is de beoogde functionaliteit?
  - Waarvoor zal de data gebruikt worden?

HO GENT

Fase 1 = Verzamelen en analyseren informatie

Voorbeeld: we willen info opslaan over films en acteurs.

- Relevante data is de titel van een film, het jaar waarin de film werd getoond, de rating-score, de namen van de acteurs …
- Er moeten films kunnen toegevoegd worden, de rating van een film moet kunnen aangepast worden …
- Het moet mogelijk zijn om een overzicht te krijgen van de films. Het moet bijvoorbeeld mogelijk zijn om te weten in welke verschillende films één specifieke acteur meespeelde, wat de gemiddelde rating is van romantische komedies, hoeveel thrillers er in 2015 werden geproduceerd …

Fase 2 = Conceptueel ontwerp

- Het conceptueel model is
  - een abstractie van de data en de onderlinge verbanden
  - formeel en ondubbelzinnig voor de databankontwerper.
  - gebruiksvriendelijk
  - doorgaans een grafische representatie
  - de basis voor communicatie en discussie tussen de gebruiker van het bedrijfsproces en de databankontwerper.
  - onafhankelijk van een databankmodel of een bestaande applicatie. Anders te vroeg gekoppeld aan dat bepaald databankmodel of die bepaalde applicatie.

HO GENT

Fase 2 = Conceptueel ontwerp

Voorbeeld: we willen info opslaan over films en acteurs.

- De data zal worden georganiseerd rond de centrale concepten 'Film' en 'Acteur'

- Gegevens die je wil opslaan
  - Film: titel, jaar waarin de film verscheen, genre, rating, aantal stemmen, adult-only, ….
  - Acteur: voornaam, familienaam, geboortejaar, jaar van overlijden

- Voorbeeld van informatie die niet in een conceptueel model opgenomen is:
  - Het geboortejaar van een acteur moet kleiner zijn dan het jaar van overlijden;
  - Van elke acteur moet het geboortejaar bekend zijn.
  - De rating van een film kan nooit kleiner zijn dan 0.

HO GENT

Fase 3 = Logisch ontwerp

- Type databank is bekend (relationele databank, NoSQL databank, hiërarchische databank, …)

- Het product zelf ligt nog niet vast
  - voor relationele databank Microsoft SQL Server of MySQL of DB2 of …
  - voor NoSQL document databank MongoDB of CouchDB of …
  - voor hiërarschische databank IMS of …

- !! Bij het opstellen van het conceptueel model en bij de overgang van het conceptueel model naar het logisch model is er mogelijk verlies van specificaties.
  → In een apart document bijhouden om te gebruiken bij de applicatie-ontwikkeling.

HO GENT

Fase 3 = Logisch ontwerp

Voorbeeld: we willen info opslaan over films en acteurs.

- We kiezen voor het relationeel databasemodel.
- De centrale concepten ‘Film’ en 'Acteur’ worden omgezet en zullen later evolueren naar tabellen.

HO GENT

Fase 4 = Fysiek ontwerp

- Is de feitelijke implementatie van het logisch model.
- Je kiest eerst een product, ook DBMS genoemd (MySQL, Microsoft SQL Server, Oracle, …).
- Je implementeert het logisch model en zet dit om in datadefinitiecode (= DDL), die kan worden verwerkt door het DBMS.
- Technische details worden toegevoegd (datatypes van de attribuuttypes, …)
- Indien mogelijk worden ook de functionele beschrijvingen ‘vertaald’ naar databaseconcepten. Zo kunnen de bedrijfsregels rond correct geboortejaar en jaar van overlijden omgezet worden naar een integriteitsrestrictie.
- DBA kan ook aanbevelingen doen in verband met de performantie.
  → zie Relational Databases and Datawarehousing (2TI)

HO GENT

Fases in Databank Ontwerp

Verzamelen en analyseren van de vereisten

- domeinanalyse
- functionele analyse
- behoefteanalyse

Conceptueel ontwerp

- conceptueel model (bijvoorbeeld EER – diagram)
- functionele beschrijving

databasemodel-onafhankelijk

Logisch ontwerp

- logisch databankschema (bijvoorbeeld relationeel)
- gedragsspecificaties

dbms-onafhankelijk

Fysiek ontwerp

- DDL-scripts
- implementatie van gedrag

HO GENT

Bron: Principes van Databases (G. De Tré)

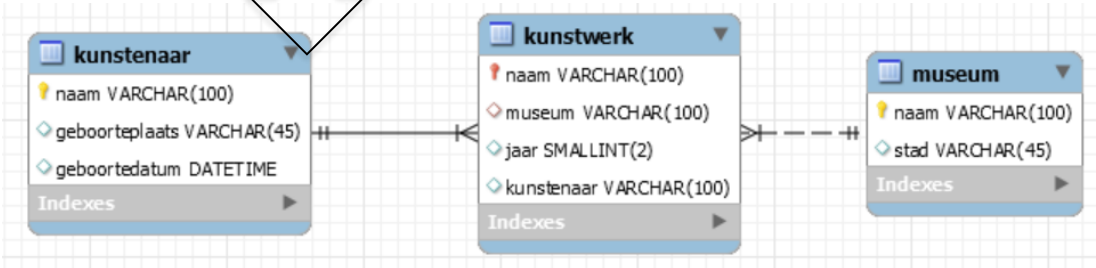
Fasen in Databank Ontwerp.

Conceptueel model

We vertalen het conceptueel model naar een relationele databank: Welke tabellen zijn er nodig? Welke kolommen? Naast relationele databanken, bestaan er ook nog andere.

Logisch model

We voorzien scripts om de databank fysisch te creëren.



Fysisch model

```
CREATE TABLE `kunstenaar` (
  `naam` varchar(100) NOT NULL,
  `geboorteplaats` varchar(45) DEFAULT NULL,
  `geboortedatum` datetime DEFAULT NULL,
  PRIMARY KEY (`naam`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

2. Entity Relationship Diagram

HO
GENT

Inleiding

- Het Entity Relationship Diagram (ERD) werd geïntroduceerd en geformaliseerd door Peter Chen in 1976.

- Het is één van de populairste voorstellingswijzen voor het conceptueel gegevensmodel.

- Een Entity Relationship Diagram heeft de volgende bouwstenen:
  - Entiteittypes
  - Attribuuttypes
  - Relatietypes

HO GENT

Entiteittype

- Een **entiteittype**
  - bestaat in de reële wereld.
  - kan zowel abstract (een tentoonstelling, firma, cursus, job, …) als fysiek (schilderij, persoon, auto, huis, …) zijn.
  - is ondubbelzinnig gedefinieerd voor een bepaalde groep gebruikers.
  - karakteriseert een collectie van entiteiten
  - heeft een naam en inhoud en is identificeerbaar.
- Een entiteit is een instantie van een entiteittype.
- In het conceptueel model nemen we entiteittypes op (geen individuele entiteiten).

HO GENT

Attribuuttype

- Een attribuuttype
  - is een karakteristiek van een entiteittype
  - beschrijft het entiteittype

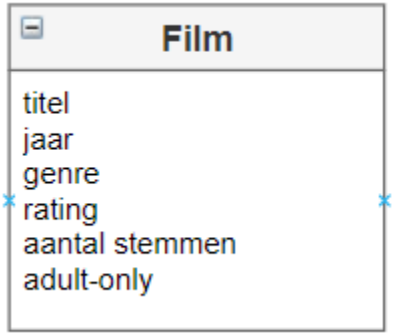
- Elke entiteit heeft een specifieke waarde voor elke attribuuttype.

HO GENT

Entiteittype en attribuuttype

Voorbeeld: visualisatie van de entiteittypes voor Film en Acteur.

- Over de ondergang van de Titanic werd in 1997 een romantische film uitgebracht die op imdb een rating van 7,9/10 haalt. Deze rating is gebaseerd op 1,2M stemmen.
- Leonardo DiCaprio (°1974) is een entiteit van het entiteittype Acteur.



HO GENT

Entiteittype en attribuuttype

- Voor een onervaren databaseontwerper kan het onduidelijk zijn of een gegeven concept al dan niet als entiteittype moet worden gemodelleerd.

- Een entiteittype is **identificeerbaar** en moet **een inhoud** hebben.

HO GENT

Attribuuttype

Het ER-model kent een aantal mogelijkheden om attribuuttypes verder te karakteriseren:

- Enkelvoudige versus samengestelde attribuuttypes
- Enkelwaardige versus meerwaardige attribuuttypes
- Afgeleide attribuuttypes
- Kandidaatsleutelattribuuttypes

HO GENT

Attribuuttype

Enkelvoudige versus samengestelde attribuuttypes

- Samengesteld attribuuttype: het attribuuttype kan nog opgesplitst worden. Bijvoorbeeld het attribuuttype ‘adres’ kan samengesteld zijn uit een ‘straat’, een ‘nummer’, een ‘postcode’ en een 'woonplaats'.
  Wij werken in het conceptueel model steeds op het niveau van enkelvoudige attribuuttypes.
- Afhankelijk van de context zullen attribuuttypes soms verder opgesplitst worden of niet. Bijvoorbeeld als het niet belangrijk is dat 'straat' of 'woonplaats' afzonderlijk moet gekend zijn, dan wordt 'adres' een enkelvoudig attribuuttype. In dat geval kan niet met de afzonderlijke delen (straat, stad, …) gewerkt worden.

HO GENT

Attribuuttype

Enkelwaardige versus meerwaardige attribuuttypes

- Enkelwaardig attribuuttype: het attribuuttype heeft één waarde. Bijvoorbeeld het attribuuttype ‘titel’ van ‘Film’ en de attribuuttypes ‘geboortejaar’ en ‘jaar overlijden’ van ‘Acteur’.
- Meerwaardig attribuuttype: het attribuuttype kan (meerdere) waarden bevatten. Bijvoorbeeld een 'Film' kan meerdere genres hebben. In dat geval is 'genre' een meerwaardig attribuuttype.
- In een ERD mogen beide voorkomen (zie later).
  Binnen deze cursus vermijden we meerwaardige attributen in het ERD.

HO
GENT

Attribuuttype

Afgeleide attribuuttypes

- De waarde van een afgeleid attribuuttype kan berekend worden op basis van waarden van andere attribuuttypes.
- Een typisch voorbeeld is ‘leeftijd’: de waarde kan berekend worden als het verschil van de huidige datum en de waarde van het attribuuttype ‘geboortedatum’
- Afgeleide attribuuttypes worden **niet opgeslagen** in de databank, omdat dit een potentiële bron van inconsistentie kan zijn.
  Dit wordt vervangen door de **basisinformatie** waaruit de waarde van het attribuuttype kan berekend worden.

HO GENT

Attribuuttype

Kandidaatsleutelattributen

- Één attribuut of meerdere attributen samen die de entiteiten van een entiteittype op een unieke, irreducibele manier identificeren, vormen een **kandidaatsleutel** van het entiteittype.
  Irreducibiliteit wil zeggen dat er geen uniciteit mag gelden als men een 1 of meerdere attributen weglaat.
- De attributen die deel uitmaken van een kandidaatsleutel noem je **kandidaatsleutelattributen**.
- Er kunnen meerdere **kandidaatsleutels** zijn. Later wordt uit de kandidaatsleutels één sleutel gekozen als primaire sleutel.

HO GENT

Attribuuttype

Kandidaatsleutelattributen

Voorbeeld: Stel er bestaan geen twee acteurs met eenzelfde combinatie van voornaam en familienaam => voornaam en familienaam zijn **kandidaatsleutelattributen** en vormen (samen) de kandidaatsleutel.

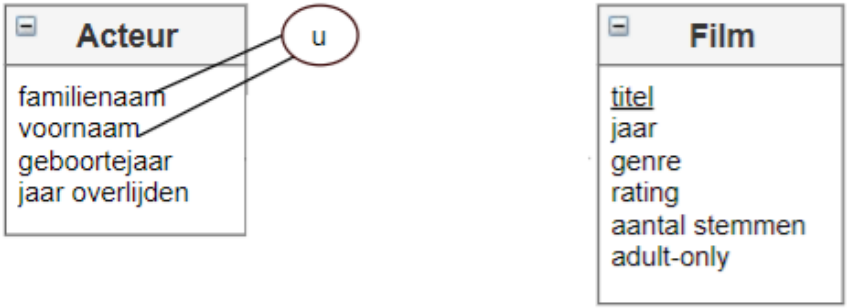
De combinatie van voornaam, familienaam en geboortedatum is geen kandidaatsleutel want de combinatie van voornaam en familienaam op zich is al uniek => voornaam, familienaam én geboortedatum zijn niet irreducibel, maw geboortedatum is overbodig om uniciteit af te dwingen.

HO GENT

Attributen

Kandidaatsleutelattributen

- Alle enkelvoudige kandidaatsleutels (bestaande uit 1 attribuuttype) worden onderlijnd.
- Indien een kandidaatsleutel uit meerdere attribuuttypes bestaat (samengestelde kandidaatsleutel), duiden we dit dan aan met de 'u'-constraint (= unique constraint).



HO GENT

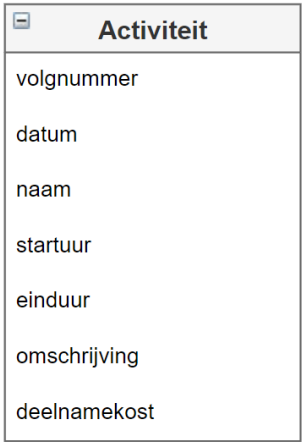
Oefening Activiteiten

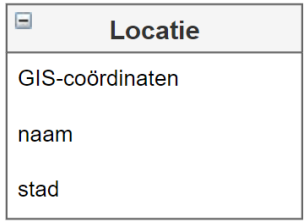
- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een deelnamekost.
- Elke activiteit start en eindigt op een bepaalde locatie. Een locatie heeft unieke GIS-coördinaten, een naam en een stad.
- Welke entiteittypes en attribuuttypes herken je?

HO GENT

Oefening Activiteiten

- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een deelnamekost.
- Elke activiteit start en eindigt op een bepaalde locatie. Een locatie heeft unieke GIS-coördinaten, een naam en een stad.
- Welke entiteittypes en attribuuttypes herken je?





HO GENT

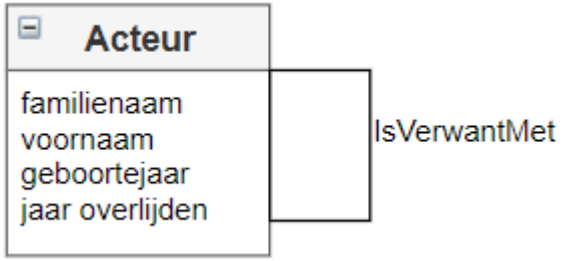
Relatietype

- Entiteittypes kunnen onderlinge verbanden hebben:
  - een acteur speelt mee in een film
  - een student volgt een aantal cursussen
- Er kunnen één, twee, drie of meer entiteittypes betrokken zijn in een relatie
  - één entiteittype: een acteur kan verwant zijn aan een andere acteur
  - twee enteittypes: in een film spelen één of meerdere acteurs
  - drie entiteittypes: een arts schrijft een medicijn voor aan een patiënt.
- Een **relatietype** is een verzameling van relaties tussen instanties van één, twee of meer al dan niet verschillende entiteittypes. Men spreekt respectievelijk van een unair (n = 1), binair (n = 2), ternair (n = 3) of n-air (n > 3) relatietype.
  Elk relatietype wordt gekenmerkt door een naam.
- In deze cursus beperken we ons tot unaire en binaire relatietypes.

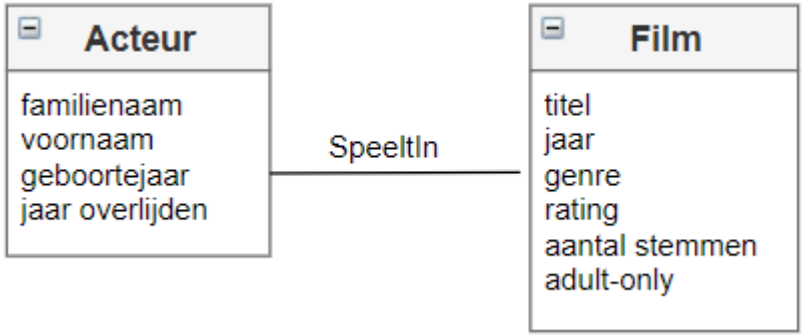
HO GENT

Relatietype

- De **graad** van een relatietype = het aantal verschillende entiteittypes die deelnemen aan het relatietype
- Voorbeeld van een unaire of recursieve relatie (Er is één entiteittype betrokken in de relatie, namelijk Acteur):

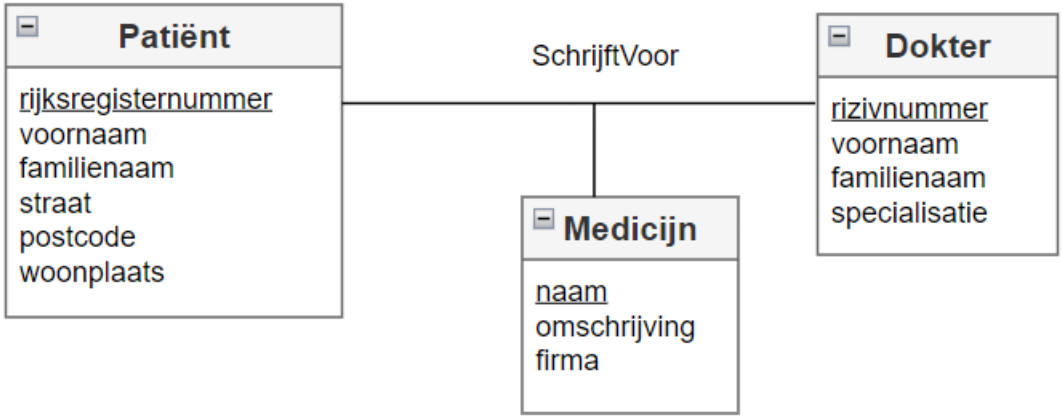


- Voorbeeld van een binaire relatie (Er zijn twee entiteittypes betrokken in de relatie, namelijk Acteur en Film):



Relatietype

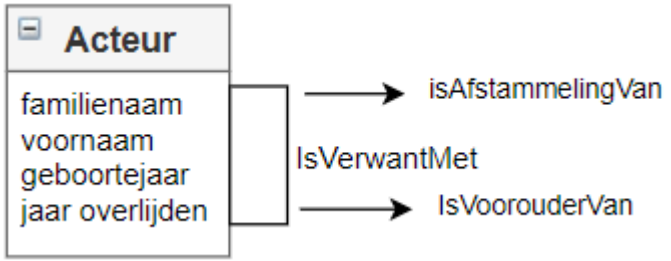
- Voorbeeld van een ternaire relatie (Er zijn 3 entiteittypes betrokken in de relatie, namelijk Patiënt, Dokter en Medicijn):



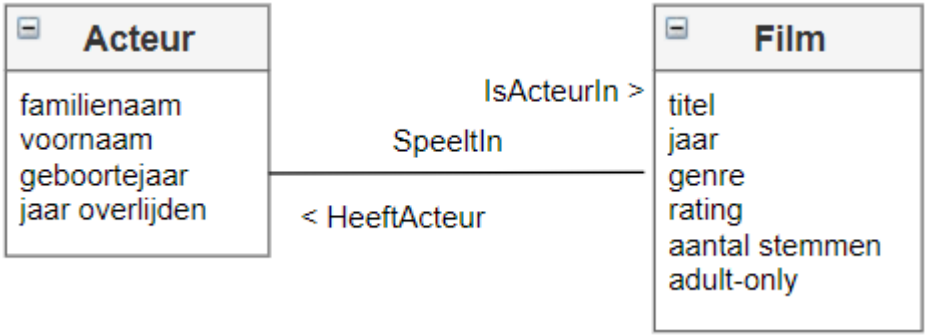
HO GENT

Relatietype

- De **rollen** van een relatietype beschrijven de specifieke rol van elk entiteittype in het relatietype.
- Voorbeeld van een unaire of recursieve relatie:



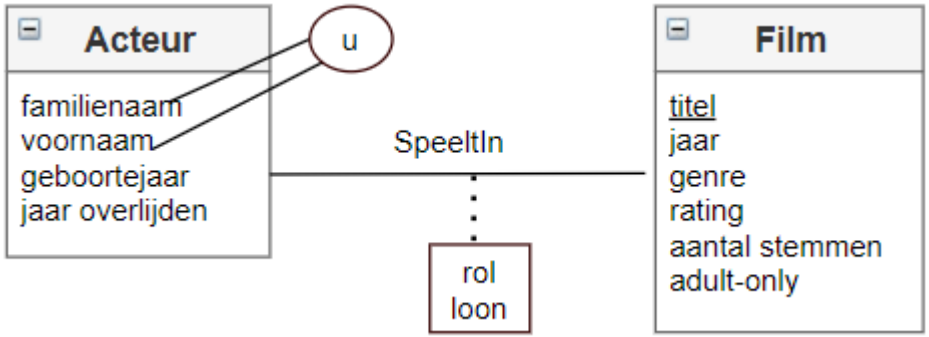
- Voorbeeld van een binaire relatie:



HO GENT

Relatie-attribuut

- Ook relatietypes kunnen eigenschappen hebben: wanneer een kenmerk een eigenschap is van het relatietype en niet van één van de betrokken entiteittypes. We spreken van een **relatie-attribuut**.
- **Voorbeeld**: de rol die een acteur vertolkte in een film. Deze rol hoort bij de 'SpeeltIn'-relatie tussen Acteur en Film en is geen eigenschap van Acteur (die verschillende rollen kan gespeeld hebben) of van Film (er zijn meerdere rollen in een film).
- Idem voor het loon dat een acteur gekregen heeft voor het meespelen in een film.



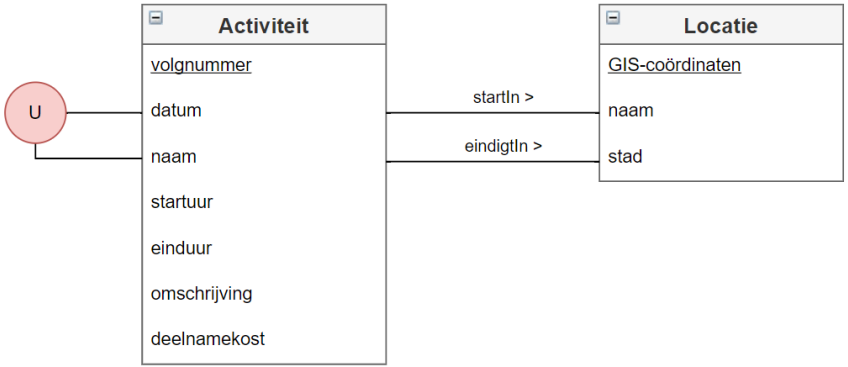
Oefening Activiteiten

- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een kostprijs om deel te nemen. De combinatie van datum en naam is ook uniek.
- Elke activiteit start en eindigt op een bepaalde locatie.
  Een locatie heeft unieke GIS-coördinaten,
  een naam en een stad.
- Geef de kandidaatsleutels

HO GENT

Oefening Activiteiten

- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een kostprijs om deel te nemen. De combinatie van datum en naam is ook uniek.
- Elke activiteit start en eindigt op een bepaalde locatie.
  Een locatie heeft unieke GIS-coördinaten,
  een naam en een stad.
- Geef de relatietype(s)



HO GENT

Cardinaliteiten

- Elk relatietype heeft een minimum- en een maximumcardinaliteit.
- Cardinaliteit betekent aantal en wordt uitgedrukt als een getal.
- De cardinaliteiten moeten afgetoetst worden met de opdrachtgever! Deze zijn vaak afhankelijk van de bedrijfsregels.
  Modelleer enkel wat je weet. We veronderstellen niets!

Verkeerde cardinaliteiten kunnen leiden tot minder kwalitatieve applicaties.

HO GENT

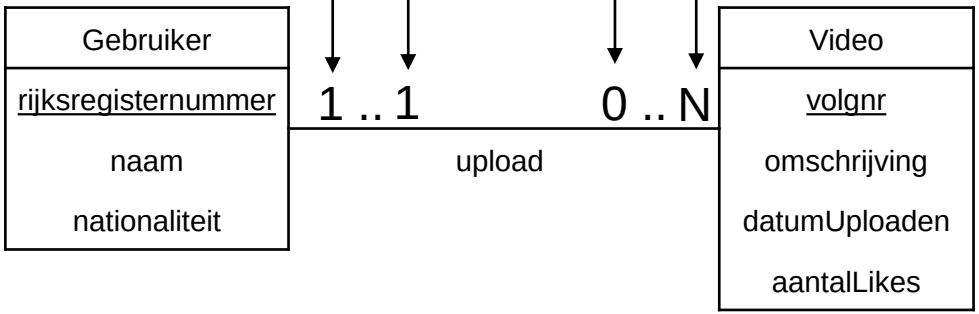
Cardinaliteiten

- Maximumcardinaliteit = het maximum aantal entiteiten van het entiteittype dat op een gegeven tijdstip **kan** deelnemen aan een relatie van het relatietype. Mogelijke waarden zijn 1 of N.
  - 1: één entiteit kan in relatie staan met maximum 1 (andere) entiteit via dit relatietype
  - N: één entiteit kan in relatie staan met N (andere) entiteiten via dit relatietype. N is een willekeurig geheel getal groter dan 1.
- Minimumcardinaliteit = het minimum aantal entiteiten van het entiteittype dat op elk tijdstip **moet** voorkomen in een relatie van het relatietype. Mogelijke waarden zijn 0 of 1.
  - 0: sommige entiteiten nemen niet deel aan de relatie. De relatie is optioneel voor dat entiteittype.
  - 1: een entiteit moet altijd in relatie staan met minimum één andere entiteit

HO GENT

Cardinaliteiten

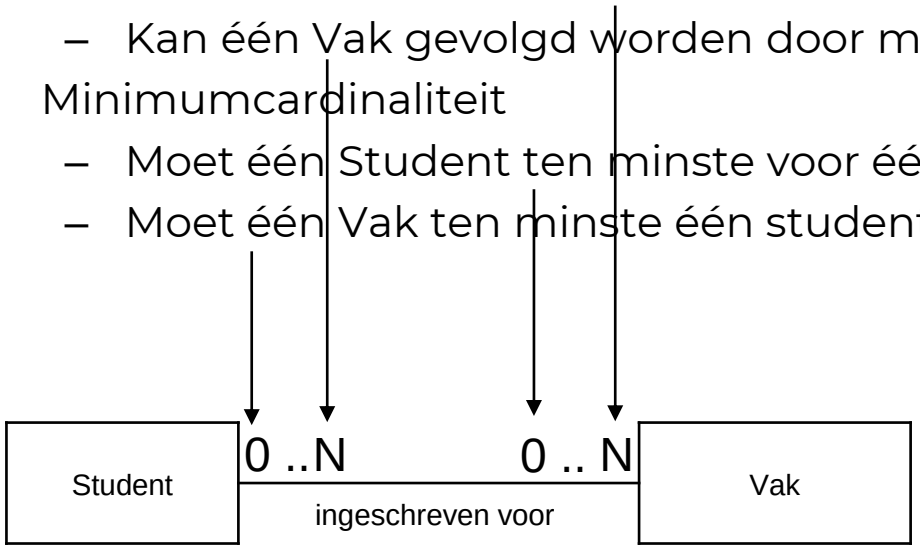
- Maximumcardinaliteit
  - Kan één Gebruiker meer dan één Video geüpload hebben? Ja => N
  - Kan één Video geüpload zijn door meer dan één Gebruiker? Neen => 1
- Minimumcardinaliteit
  - Moet één Gebruiker ten minste één Video geüpload hebben? Neen => 0
  - Moet één Video geüpload zijn door ten minste één Gebruiker? Ja => 1



HO GENT

Cardinaliteiten

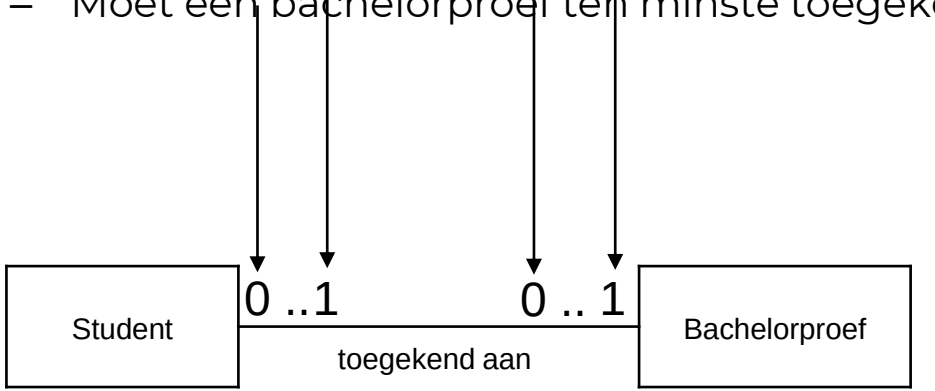
- Maximumcardinaliteit
  - Kan één Student ingeschreven zijn voor meer dan één Vak ? Ja => N
  - Kan één Vak gevolgd worden door meer dan één Student? Ja => N
- Minimumcardinaliteit
  - Moet één Student ten minste voor één Vak ingeschreven zijn? Neen => 0
  - Moet één Vak ten minste één student hebben? Neen => 0



HO GENT

Cardinaliteiten

- Maximumcardinaliteit
  - Kan één Student toegekend zijn aan meer dan één bachelorproef? Neen => 1
  - Kan één bachelorproef toegekend zijn aan meer dan één Student? Neen => 1
- Minimumcardinaliteit
  - Moet één Student toegekend zijn aan ten minste één bachelorproef? Neen => 0
  - Moet één bachelorproef ten minste toegekend zijn aan één Student? Neen => 0

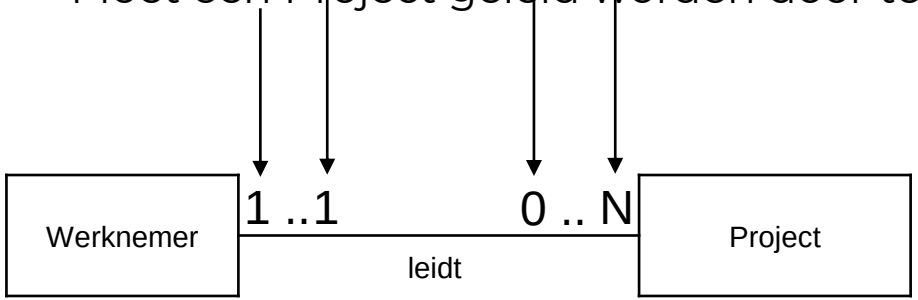


HO GENT

Cardinaliteiten

Voorbeeld van bedrijfsspecifieke cardinaliteiten.

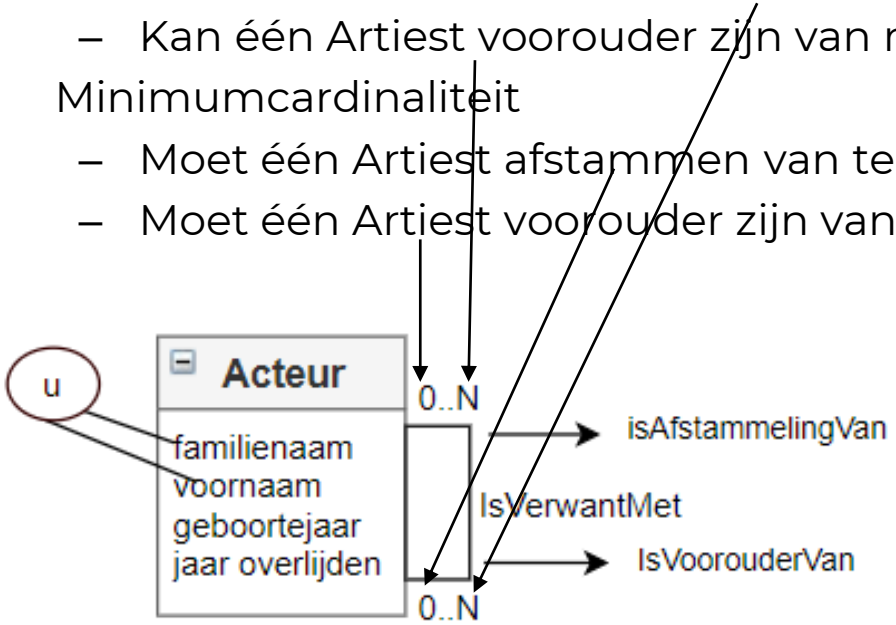
- Maximumcardinaliteit
  - Kan één Werknemer meer dan één Project leiden? Ja => N
  - Kan één Project geleid worden door meer dan één Werknemer? Neen => 1
- Minimumcardinaliteit
  - Moet één Werknemer ten minste één Project leiden? Neen => 0
  - Moet één Project geleid worden door ten minste één Werknemer? Ja => 1



HO GENT

Cardinaliteiten

- Maximumcardinaliteit
  - Kan één Artiest afstammen van meer dan één Artiest? Ja => N
  - Kan één Artiest voorouder zijn van meer dan één Artiest? Ja => N
- Minimumcardinaliteit
  - Moet één Artiest afstammen van ten minste één Artiest? Neen => 0
  - Moet één Artiest voorouder zijn van ten minste één Artiest? Neen => 0



HO GENT

Cardinaliteiten

- Relatietypes worden vaak gekarakteriseerd door de maximumcardinaliteit van elk van de rollen.

- In het geval van een unaire of binaire relatie geeft dit aanleiding tot:
  - 1-op-1 relatie → 1:1
  - 1-op-veel-relatie → 1:N of N:1
  - veel-op-veel-relatie → M:N of N:N

- Een verplichte minimumcardinaliteit wijst op bestaansafhankelijkheid (zie later).

HO GENT

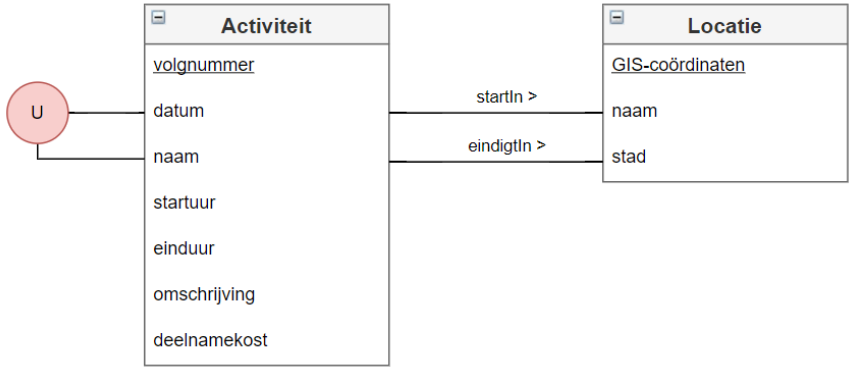
Cardinaliteiten

- Voorbeelden
  - Een student kan voor minimaal 0 en voor maximaal M vakken ingeschreven zijn.
    Voor een vak kunnen minimaal 0 en maximaal N studenten ingeschreven zijn.
    → M:N (een veel-op-veel-relatie)
  - Een bachelorproef kan aan minimaal 0 en aan maximaal 1 student toegekend zijn.
    Een student kan minimaal 0 en maximaal 1 bachelorproef uitwerken.
    → 1:1 (een één-op-één-relatie)
  - Een project wordt geleid door juist 1 werknemer.
    Een werknemer kan min 0 en max N projecten leiden.
    → 1:N (een één-op-veel-relatie)

HO GENT

Oefening Activiteiten

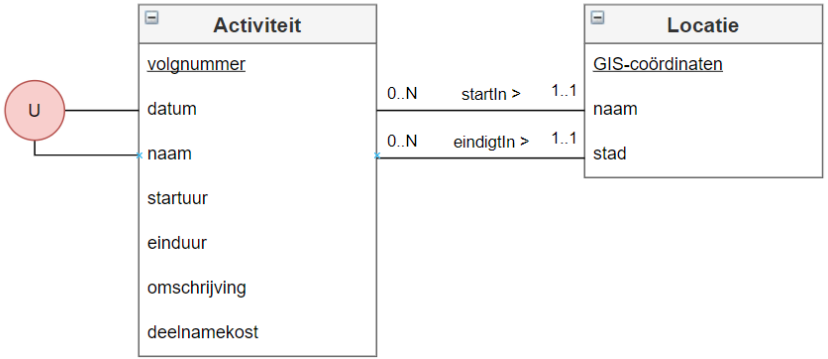
- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een kostprijs om deel te nemen. De combinatie van datum en naam is ook uniek.
- Elke activiteit start en eindigt op een bepaalde locatie.
  Een locatie heeft unieke GIS-coördinaten, een naam en een stad.
- Geef de cardinaliteiten



HO GENT

Oefening Activiteiten

- Je wil een activiteitendatabank creëren
- Elke activiteit krijgt een uniek volgnummer, een datum, een naam, een startuur, vermoedelijk einduur, een korte omschrijving en een kostprijs om deel te nemen. De combinatie van datum en naam is ook uniek.
- Elke activiteit start en eindigt op een bepaalde locatie.
  Een locatie heeft unieke GIS-coördinaten, een naam en een stad.
- Geef de cardinaliteiten

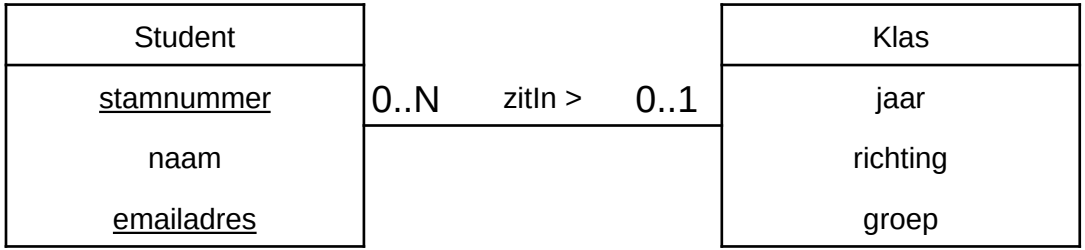


HO GENT

3. Oefeningen

HO
GENT

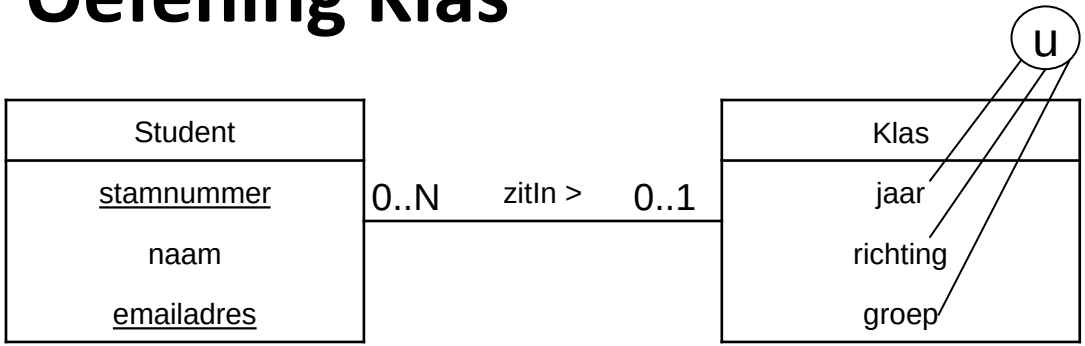
Oefening Klas



- Een student kan worden geïdentificeerd aan de hand van zijn stamnummer of aan de hand van zijn e-mailadres. Een student heeft een naam.
- Een klas heeft niet verplicht meerdere studenten.
- Een student zit maximaal in 1 klas.
- De klasgroepen zijn genummerd per jaar en richting

HO GENT

Oefening Klas

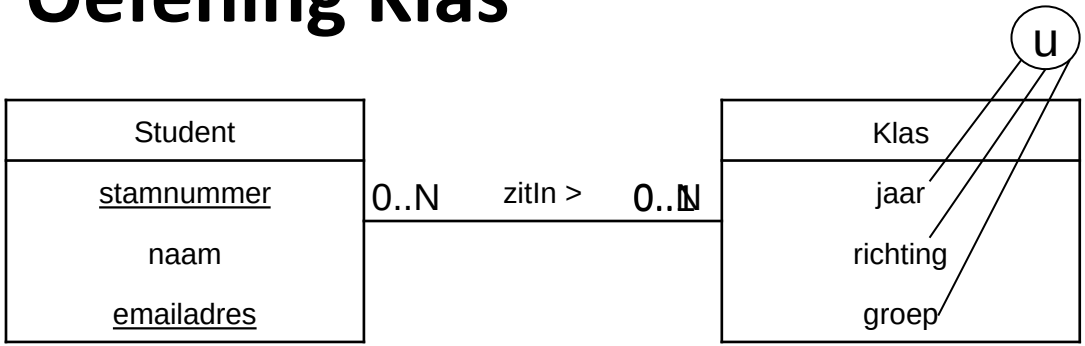


Pas het ERD aan

- Een klas wordt uniek geïdentificeerd aan de hand van de combinatie van jaar, richting en groep.

HO GENT

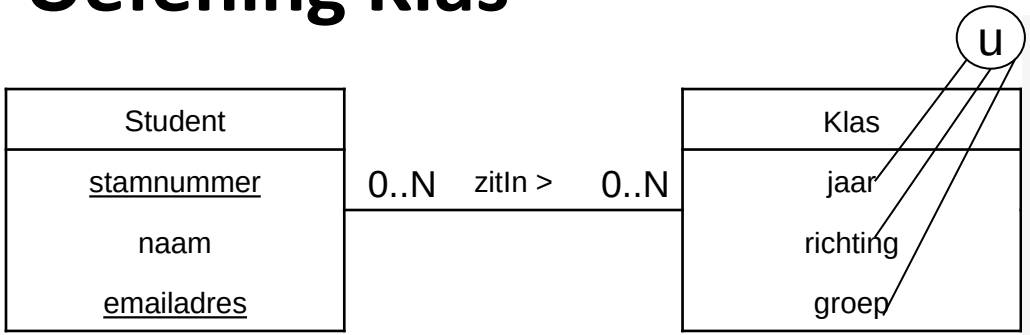
Oefening Klas



Pas het ERD aan

- Een student kan in meerdere klassen zitten.

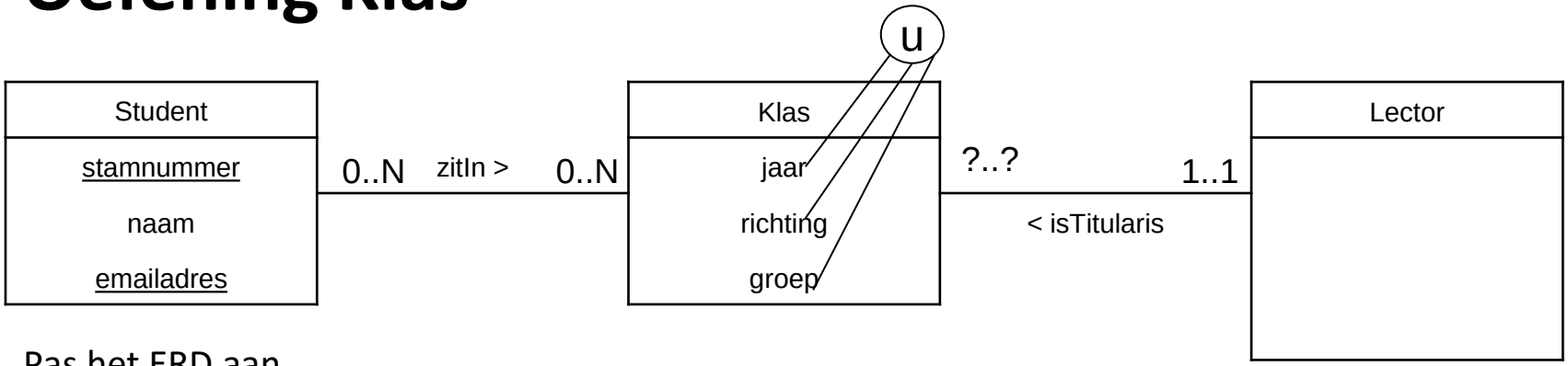
Oefening Klas



Pas het ERD aan

- Elke klas heeft juist één lector als titularis.

Oefening Klas

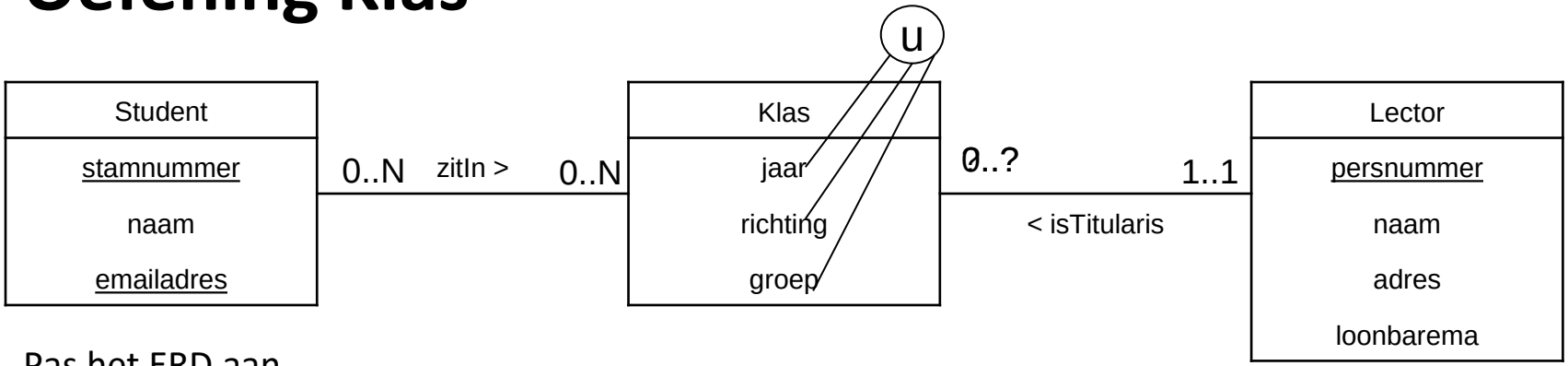


Pas het ERD aan

- Van een lector worden een uniek personeelsnummer, naam, adres en loonbarema bijgehouden

HO GENT

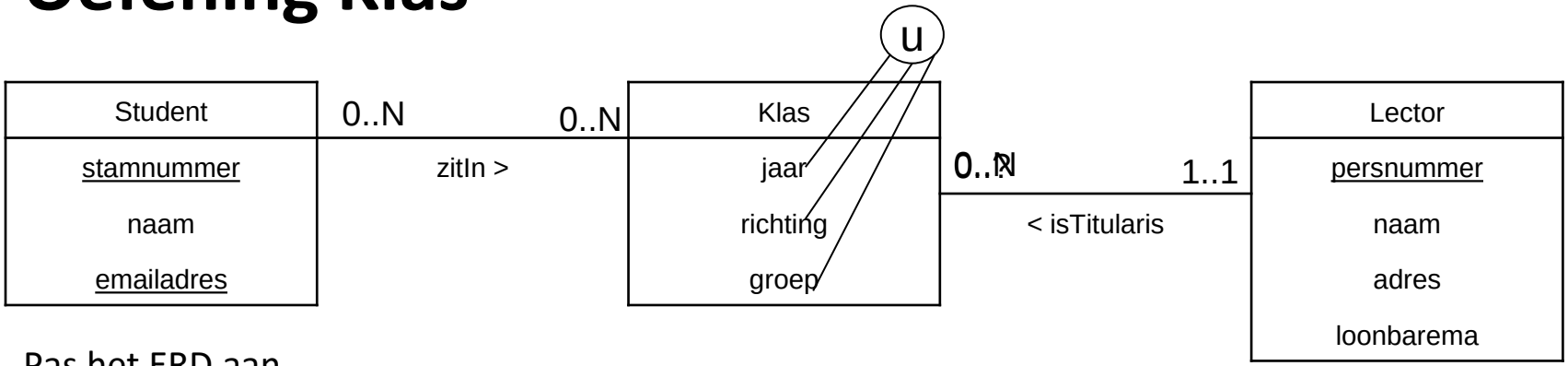
Oefening Klas



Pas het ERD aan

- Een lector hoeft geen titularis te zijn

Oefening Klas



Pas het ERD aan

- Een lector kan titularis zijn van meerdere klassen

Oefening Bibliotheek

- Een bibliotheek wil een databank ontwerpen voor het bijhouden van informatie over de boeken die er aanwezig zijn.
- Van elk boek is maar 1 exemplaar aanwezig in de bibliotheek.
- Er moet kunnen opgevraagd worden of een boek aanwezig is of niet en zo ja in welke rij het kan gevonden worden.
- Van elk boek moet volgende info kunnen opgevraagd worden: ISBN (unieke identificatie van een boek), titel, auteur(s), aantal blz, prijs, korte inhoud.
- Elk boek heeft minstens 1 auteur. Van die auteur kennen we een unieke naam.

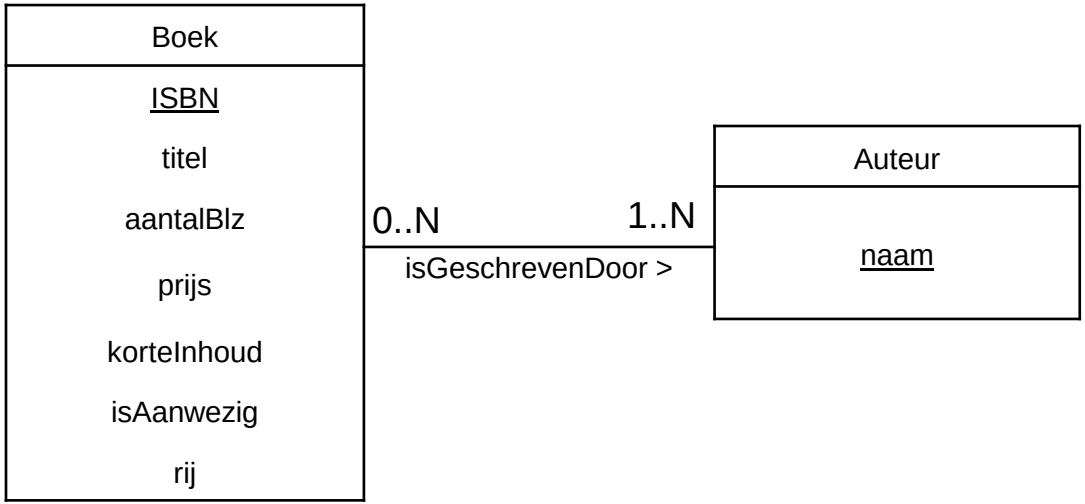
HO GENT

Oefening Bibliotheek

Boek
ISBN
titel
auteurs
aantalBlz
prijs
korteInhoud
isAanwezig
rij

HO GENT

Oefening Bibliotheek



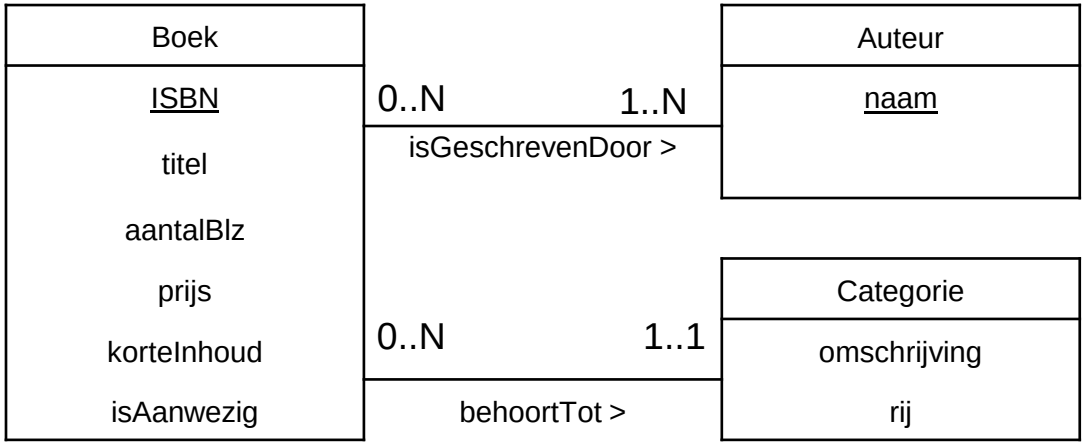
HO GENT

Oefening Bibliotheek

Breid het ERD uit:

- Elk boek behoort tot een bepaalde categorie (historische roman, thriller, ….).
- Elke categorie heeft een eigen plaats (rij) in de bibliotheek.

Oefening Bibliotheek



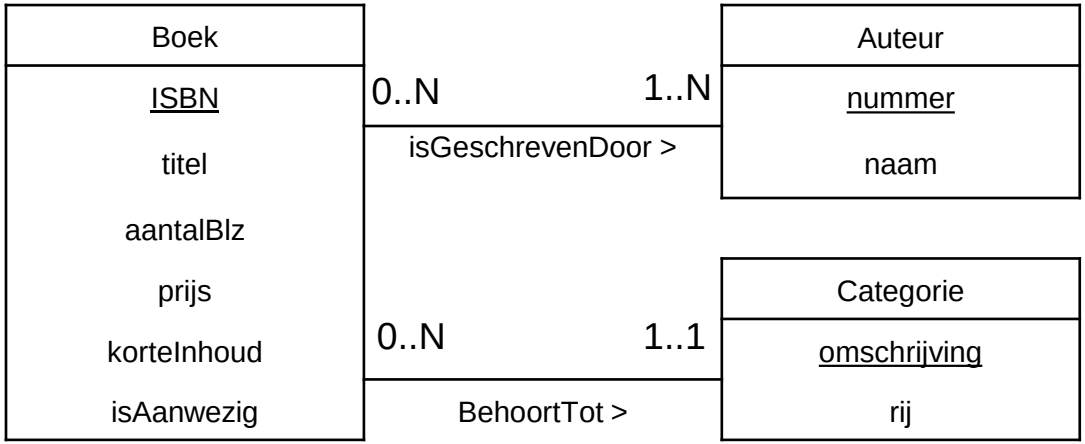
HO GENT

Oefening Bibliotheek

- Elk entiteittype moet zich kunnen identificeren.
- Hier hebben we geen identifier voor Categorie.
- Als ontwerper mag je nooit zelf iets beslissen maar moet je overleggen met de opdrachtgever!
- Na overleg blijkt dat
  - een categorie geïdentificeerd wordt aan de hand van zijn omschrijving
  - naam van een auteur is geen goede identificatie → een oplopend nummer
- Pas het ERD aan!

HO GENT

Oefening Bibliotheek



HO GENT

4. Bronnen

HO
GENT

Bronnen

- Principles of database management (W. Lemahieu, S. Vanden Broucke, B. Baesens)
  - 3.1 + 3.2.1 + 3.2.2 + 3.2.3 + 3.2.4 + 3.2.6
- Principes van databases (G. De Tré)

HO GENT

Databases

H3 zwakke entiteiten

HO
GENT

H3. Zwakke entiteiten

1. Zwakke entiteitstypes
2. Oefeningen
3. Historiek

HO GENT

1. Zwakke entiteitstypes

HO
GENT

Herhaling

- Bouwstenen
  - Entititeittypes
  - Attribuuttypes
  - Relatietypes
    - Maximumcardinaliteit: 1 of N
    - Minimumcardinaliteit: 0 of 1

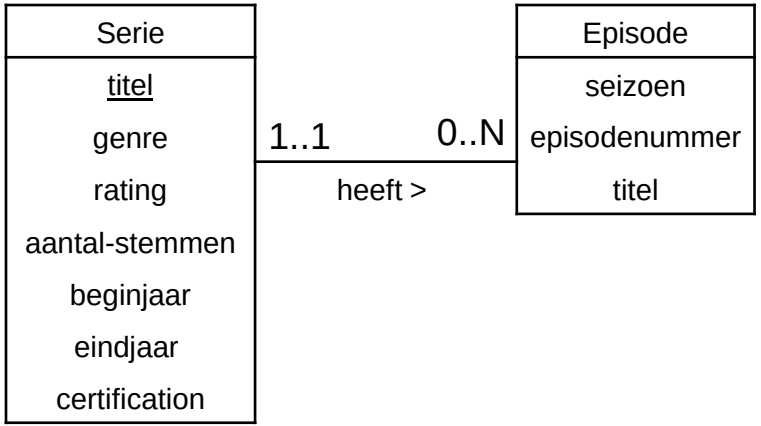
HO GENT

Inleiding

- Stel dat we in plaats van Films, informatie over Series en Acteurs willen opslaan in de databank.
- Een serie heeft een titel, een genre, een rating een aantal-stemmen, een jaartal waarin de serie startte en eventueel een jaartal waarin de laatste episode van de serie werd uitgezonden en een certification (bijvoorbeeld 16, 18+, 20, …)
- Er kunnen meerdere episodes per seizoen zijn.

HO
GENT

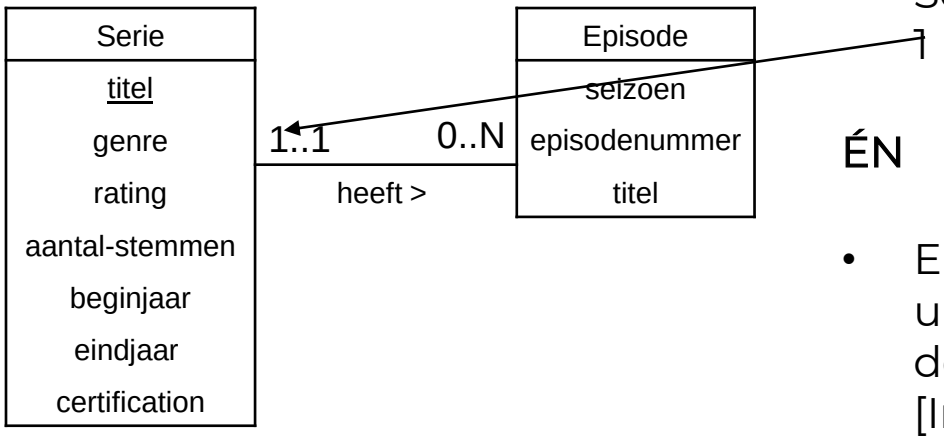
Inleiding – Poging



- Het kandidaatsleutelattribuuttype van het entiteittype Serie is: titel.
- Wat is het kandidaatsleutelattribuuttype van het entiteittype Episode?

HO GENT

Inleiding – Poging



- Episode is bestaansafhankelijk van Serie want de minimumcardinaliteit is 1

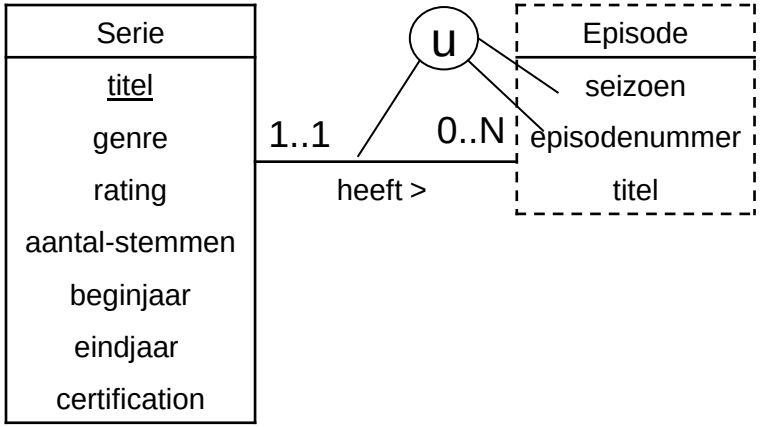
ÉN

- Episode is een entiteittype dat niet uniek kan geïdentificeerd worden aan de hand van zijn eigen attribuuttypes. [In theorie is het immers mogelijk dat 2 episodes van 2 verschillende series hetzelfde seizoen, episodenummer én dezelfde titel zouden hebben]

⇨ Episode is een **zwak entiteittype**.

HO GENT

Inleiding



- Episode is een **zwak entiteittype.**
- In een ERD wordt een zwak entiteittype aangeduid met een stippellijn.
- Voor de identificatie wordt een kandidaatsleutelattribuuttype van het zwak entiteittype gekoppeld aan **de relatie**. Dit attribuuttype wordt **niet onderlijnd**. Het is immers geen kandidaatsleutel op zich maar maakt deel uit van een kandidaatsleutel!
- Concreet voor Episode
  - Identificatie aan de hand van seizoen en episodenummer en de relatie met Serie.
  - De attribuuttypes seizoen en episodenummer worden **niet onderlijnd**!

HO GENT

Zwak entiteittype: definitie

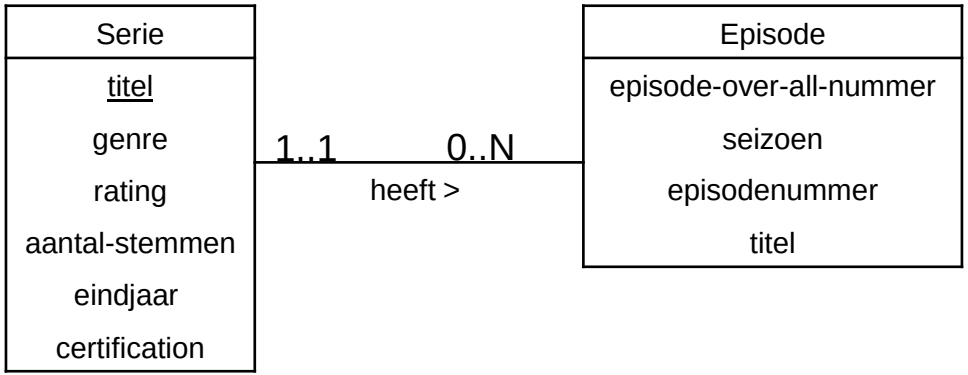
- Een zwak enteittype
  - is bestaansafhankelijk van 1 of meerdere andere entiteittypes

  ÉN
  - heeft die entiteittypes nodig om zich te kunnen identificeren

  → Deze entiteittypes noemt men de **identificerende entiteittypes** van het zwakke entiteittype.
- Zwakke entiteittypes kan je niet uniek identificeren **zonder** hun identificerende entiteittypes ⇨ zwakke entiteittypes kunnen geen kandidaatsleutel hebben.
- Zwakke entiteittypes hebben wel kandidaatsleutelattribuuttypes. Die zorgen, samen met de kandidaatsleutelattribuuttypes van de identificerende entiteittypes, voor een unieke, irreducibele identificatie van de zwakke entiteittypes.
  → de kandidaatsleutelattribuuttypes van het zwakke entiteittype vormt een **partiële kandidaatsleutel**.

HO GENT

Zwak entiteittype versus bestaansafhankelijk



- Stel: in de bedrijfsregels wordt vastgelegd dat alle Episodes oplopend genummerd worden, over alle series heen (episode-over-all-nummer).
  → het entiteittype Episode is nog steeds bestaansafhankelijk van Serie maar is niet langer zwak. Het episode-over-all-nummer kan een Episode identificeren. → **Enkel bestaansafhankelijkheid is niet voldoende om te spreken van een zwak entiteittype!**
- De specifieke bedrijfsregels bepalen of een entiteittype zwak is.

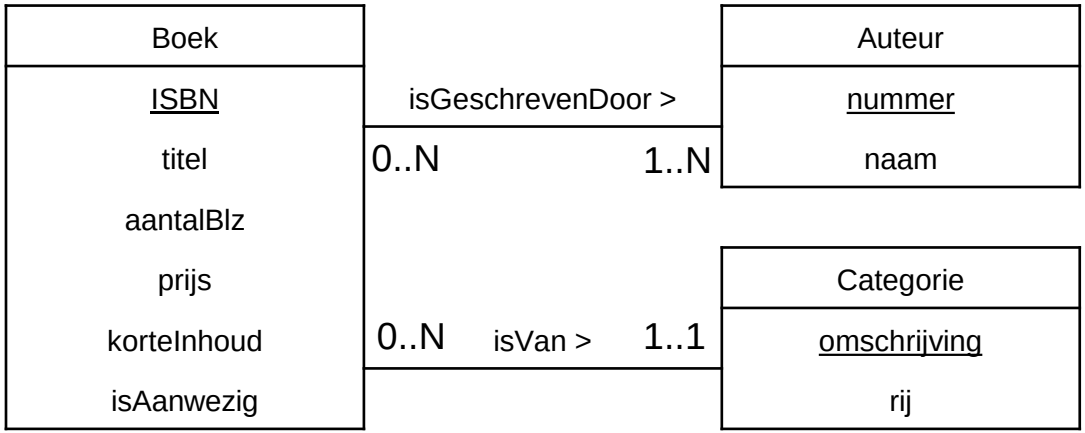
HO GENT

2. Oefeningen

HO
GENT

Oefening Bibliotheek

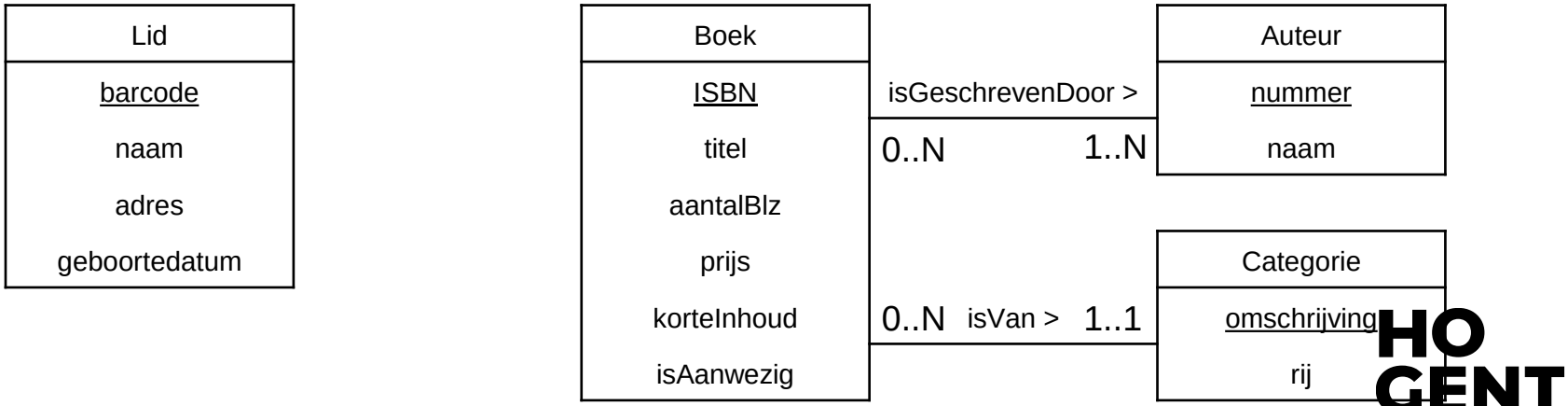
- Breid het ERD (uit vorige les) uit:
  - Enkel leden van de bib kunnen boeken ontlenen. Zij krijgen een lidkaart met unieke barcode. Van elk lid is geweten wat zijn naam, leeftijd en adres is.



HO GENT

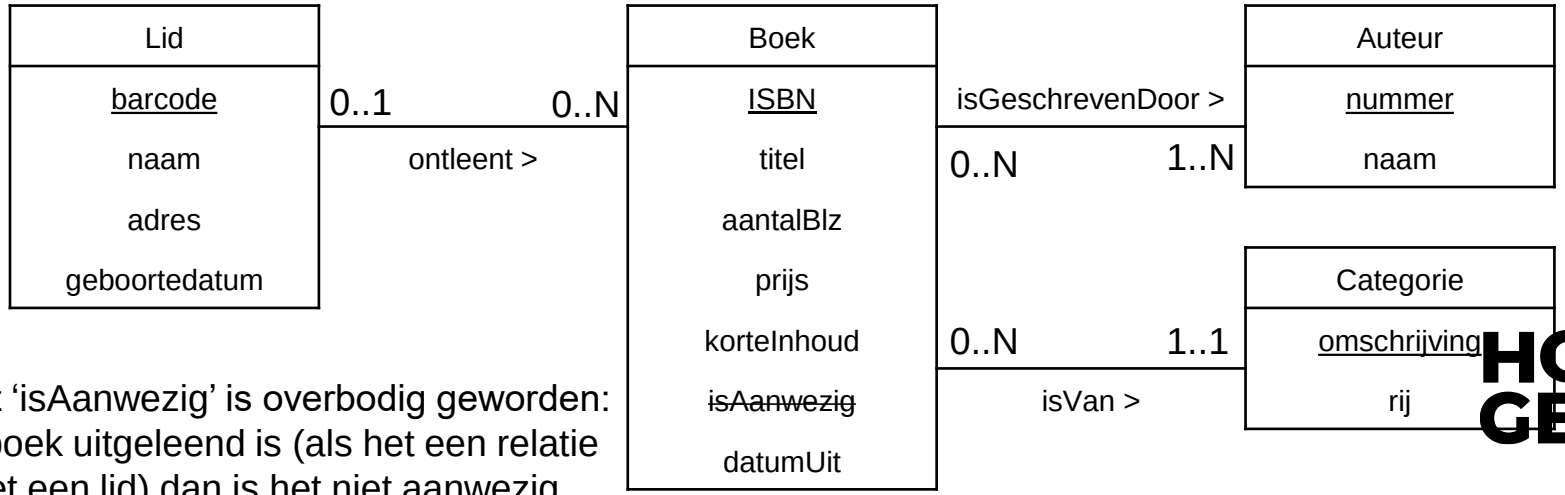
Oefening Bibliotheek

- Breid het ERD uit:
  - Telkens een lid een boek ontleent, wordt de datum van ontlening bijgehouden. Een lid kan natuurlijk meer dan één boek ontlenen.



Oefening Bibliotheek

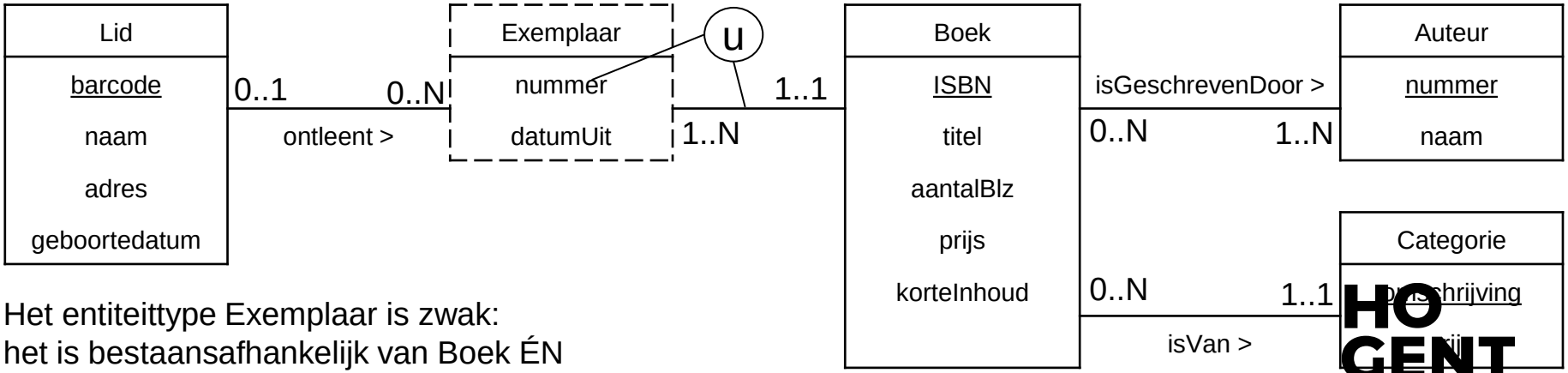
- Breid het ERD uit:
  - Van een boek kunnen meerdere exemplaren aanwezig zijn (altijd minstens 1).
    Elk exemplaar krijgt een oplopende nummering per boek.
    Het is een exemplaar van een boek dat uitgeleend wordt.



Attribuut ‘isAanwezig’ is overbodig geworden: als het boek uitgeleend is (als het een relatie heeft met een lid) dan is het niet aanwezig.

Oefening Bibliotheek

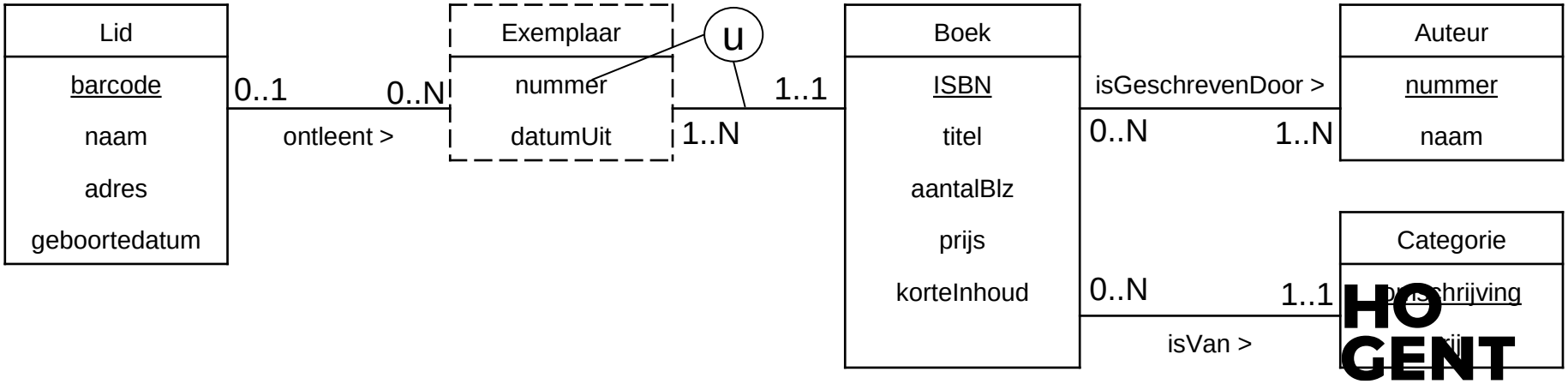
- Breid het ERD uit:
  - Een boek wordt gratis uitgeleend gedurende 20 dagen. Indien een lid het boek te laat inlevert wordt een boete van 10 cent per dag te laat aangerekend.



Het entiteittype Exemplaar is zwak:
het is bestaansafhankelijk van Boek ÉN
het heeft de relatie met Boek nodig om zich te identificeren.

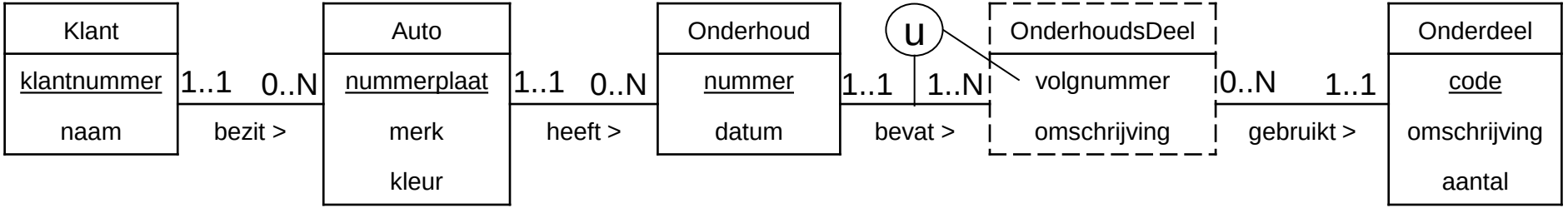
Oefening Bibliotheek

- Er verandert niets aan het ERD! Dit zijn procesgegevens.
  Wanneer de gebruiker het boek vandaag terugbrengt en
  vandaag - datumUitlening > 20 => boete = 0.10 * ((vandaag-datumUitlening) - 20)



Oefening Auto

- Bekijk het onderstaande ERD.



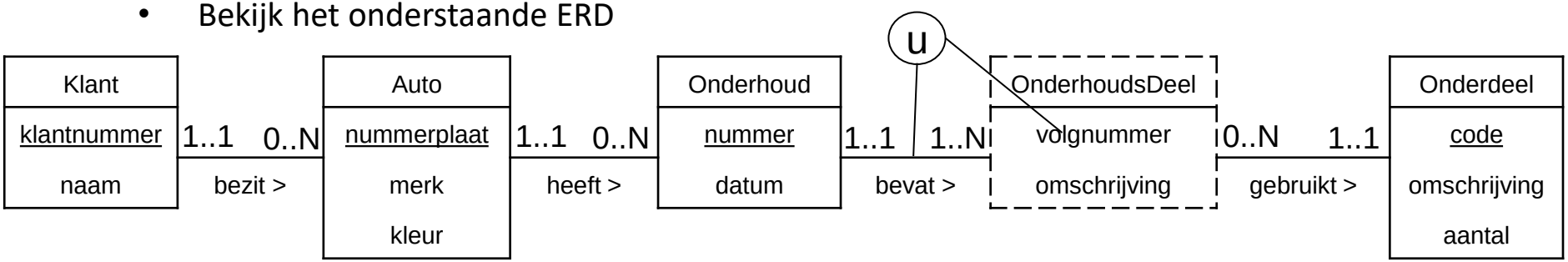
- Het is de bedoeling dat we data van de volgende vorm kunnen opslaan in de databank.

Onderhoud	Nummer Onderhoudsdeel	Omschrijving OnderhoudsDeel
10001	1	Brandstoffilter vervangen
10001	2	Luchtfilter vervangen
10001	3	Dynamoriem aanspannen
10002	1	Achterlichten vervangen
10003	1	Oliefilter vervangen
10003	2	Luchtfilter vervangen

Onderhoud	Nummer Onderhoudsdeel	Code Gebruikt Onderdeel	Aantal Gebruikt Onderdeel
10001	1	FF-015	1
10001	1	AF-1187	2
10001	2	BU-2145	1
10003	1	OF-2113	1
10003	1	QT-1578	2
10003	2	BU-2145	1

Oefening Auto

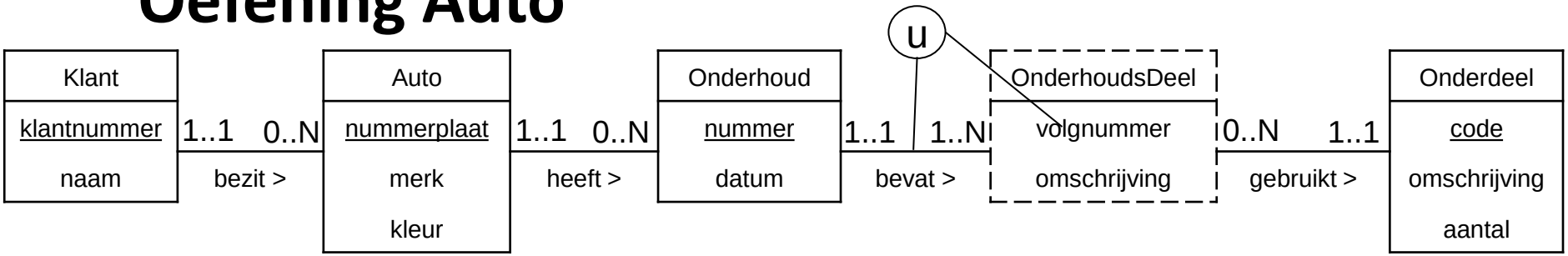
- Bekijk het onderstaande ERD



- Onderhoud is bestaansafhankelijk van Auto. Waarom?
- OnderhoudsDeel is bestaansafhankelijk van Onderdeel. Waarom?
- Onderhoud is niet zwak en Onderhoudsdeel is wel zwak. Waarom?

HO GENT

Oefening Auto



- Bepaal op basis van het bovenstaande ERD of de volgende uitspraken correct zijn of niet.
  - 1. Een klant kan meer dan één auto hebben.
  - 2. Alle klanten hebben ten minste één auto.
  - 3. Een auto is het bezit van exact één klant.
  - 4. Een auto kan meer dan één onderhoud hebben.
  - 5. Elk onderhoud behoort bij één auto.
  - 6. Alle auto's hebben ten minste één onderhoud.
  - 7. Elk onderhoud kan meer dan één onderdeel gebruiken.
  - 8. Een onderdeel kan gebruikt worden bij meer dan één onderhoud.
  - 9. Een onderhoud kan meer dan één onderhoudsDeel bevatten.

HO GENT

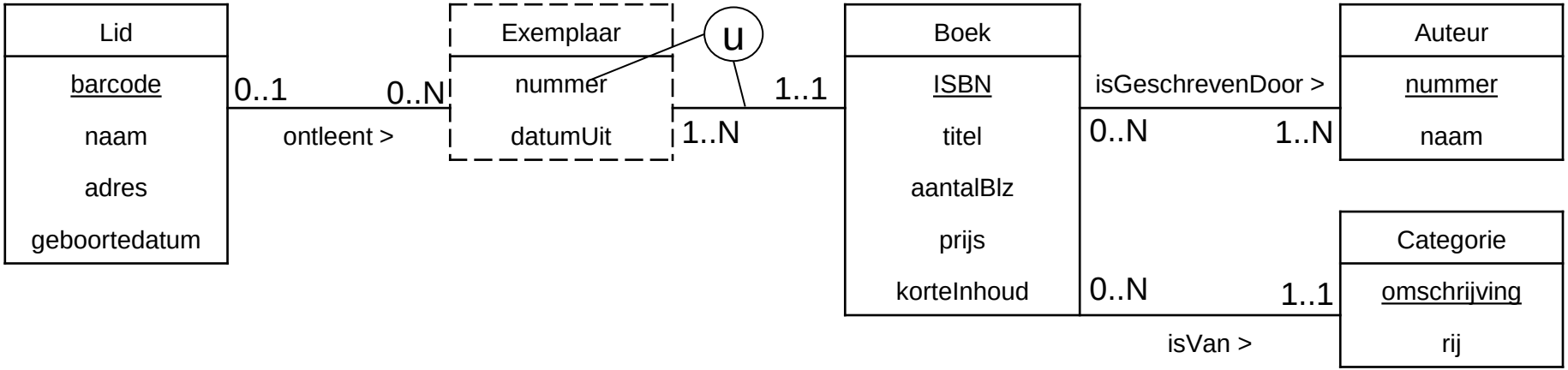
3. Historiek

Praktische toepassing van zwakke entiteiten

HO GENT

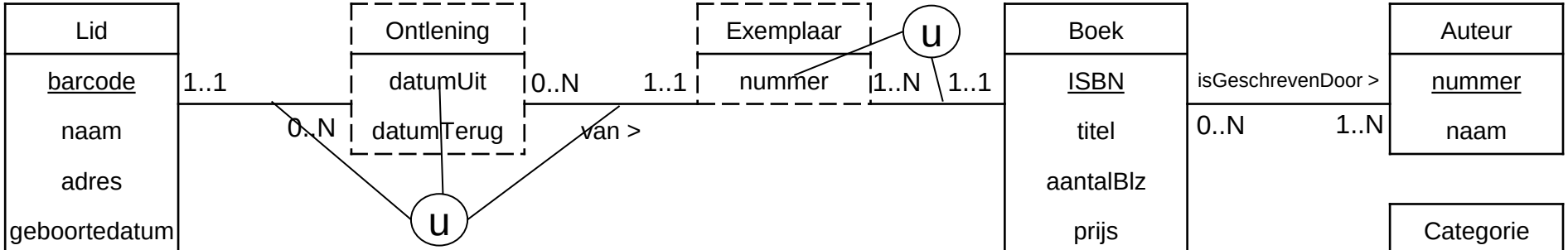
Inleiding – Oef bibliotheek

- De datum waarop een Lid een Exemplaar van een Boek uitleent, verdwijnt uit de databank wanneer het Lid dit Exemplaar terug brengt.
  Dit strookt niet met wat we gewend zijn in de realiteit.
  Soms is het noodzakelijk een overzicht van bepaalde gegevens uit het verleden te hebben en te kunnen weergeven – in dit geval de ontleningen van elk exemplaar van een boek sedert dit werd aangekocht
- Hoe kan je dit oplossen?



Historiek

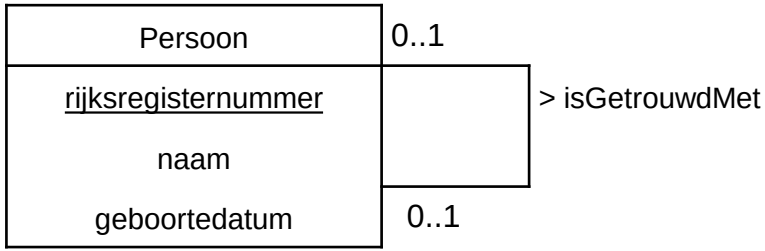
- Er wordt een bijkomend, zwak entiteittype Ontlening toegevoegd.
- Ontlening is zwak omdat het bestaansafhankelijk is van Lid en van Exemplaar ÉN omdat het zelf niet over voldoende attribuuttypes beschikt om zich te identificeren
- We noemen het entiteittype Ontlening een **historiek**.



- Een historiek wordt altijd voorgesteld door een apart, zwak entiteittype en zal altijd een datum-attribuut bevatten.

Historiek – Nog een voorbeeld

- Een persoon kan op 1 bepaald moment in de tijd maar getrouwd zijn met maximum 1 andere persoon => Maximumcardinaliteit = 1

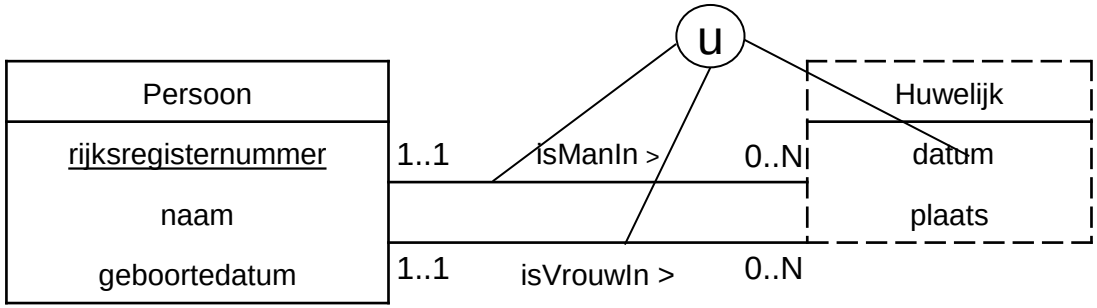


- Indien een koppel uit elkaar gaat, is in de databank geen spoor meer te vinden van het feit dat die Personen ooit met elkaar getrouwd waren.

HO GENT

Historiek – Nog een voorbeeld

- Indien je historische gegevens wil bijhouden, moet je hier gebruik maken van een Historiek.



Opmerking: de namen isManIn en isVrouwIn werden hier enkel gekozen voor de duidelijkheid.

- Op deze manier is het perfect mogelijk de geschiedenis van huwelijken van 1 Persoon met (verschillende) andere Personen te reconstrueren in de tijd.

HO GENT

Databases

H4 enhanced ERD

HO
GENT

H4. Enhanced Entity Relationship Diagram

1. Inleiding
2. Specialisatie
3. Oefeningen
4. Beperkingen van het ERD
5. Oefeningen

HO GENT

1. Inleiding

HO
GENT

Inleiding

- Om de realiteit beter te kunnen weergeven, worden de mogelijkheden voor entiteittypes verder uitgebreid.

- Het Enhanced Entity Relationship Diagram of EERD is een uitbreiding van het ERD.
  - alle concepten uit het ERD blijven behouden, namelijk entiteittype, attribuuttype en relatietype
  - een nieuw modelleringsconcept wordt toegevoegd
    - specialisatie/generalisatie

HO GENT

2. Specialisatie

HO
GENT

Specialisatie - Inleiding

- Een entiteittype is een verzameling van entiteiten met gemeenschappelijke karakteristieken.

- Het komt voor dat dergelijke collecties verder moeten worden onderverdeeld in deelverzamelingen:
  - zijn belangrijk voor de gebruikers of toepassingen,
  - moeten apart kunnen behandeld worden.

HO GENT

Specialisatie - Voorbeeld

- Aan een film of een serie werken veel meer mensen mee dan enkel acteurs. Denk bijvoorbeeld aan componisten die verantwoordelijk zijn voor de achtergrondmuziek, schrijvers die het plot bedenken, regisseurs, cameramensen, monteurs … Al deze mensen behoren tot de 'crew'.

- Entiteittype CrewMember heeft een nationaliteit, rijksregisternummer, voornaam, familienaam en een geboortedatum

Specialisatie - Voorbeeld

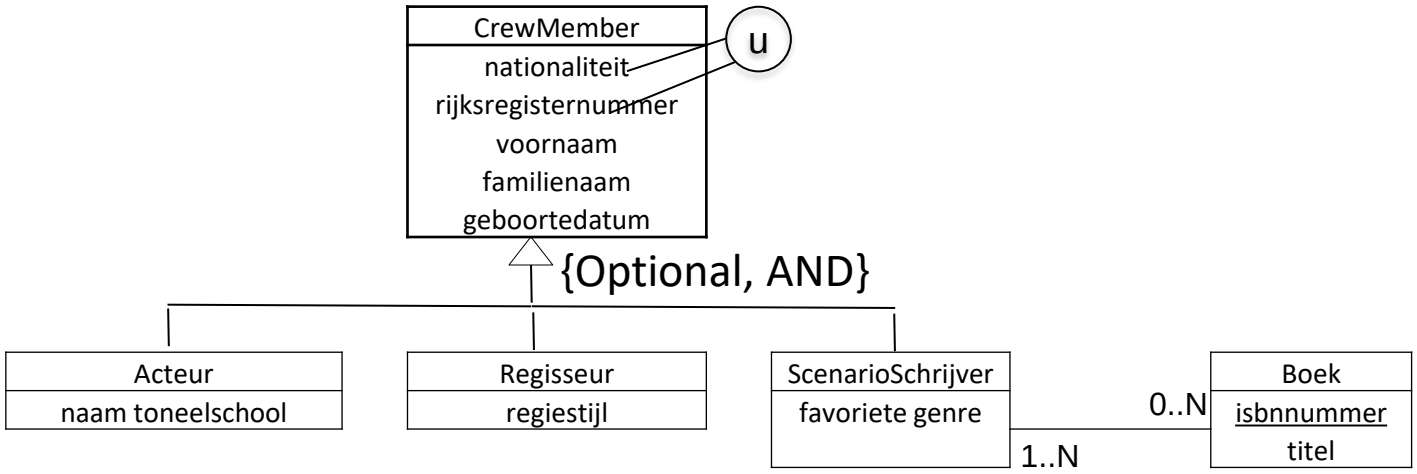
- Een Componist IS-EEN CrewMember.

  Een Acteur IS-EEN CrewMember.

  Een Schrijver IS-EEN CrewMember.
- **Het omgekeerde geldt niet** : een CrewMember is niet noodzakelijk een Componist; een CrewMember is niet noodzakelijk een Acteur …
- Een Componist, Acteur en Schrijver kunnen
  - extra attribuuttypes hebben (muziekstijl, naam toneelschool …)
  - extra relatietypes hebben (bvb. boeken die ze geschreven hebben).
- Componist, Acteur EN Schrijver erven **alle** attribuuttypes van CrewMember.

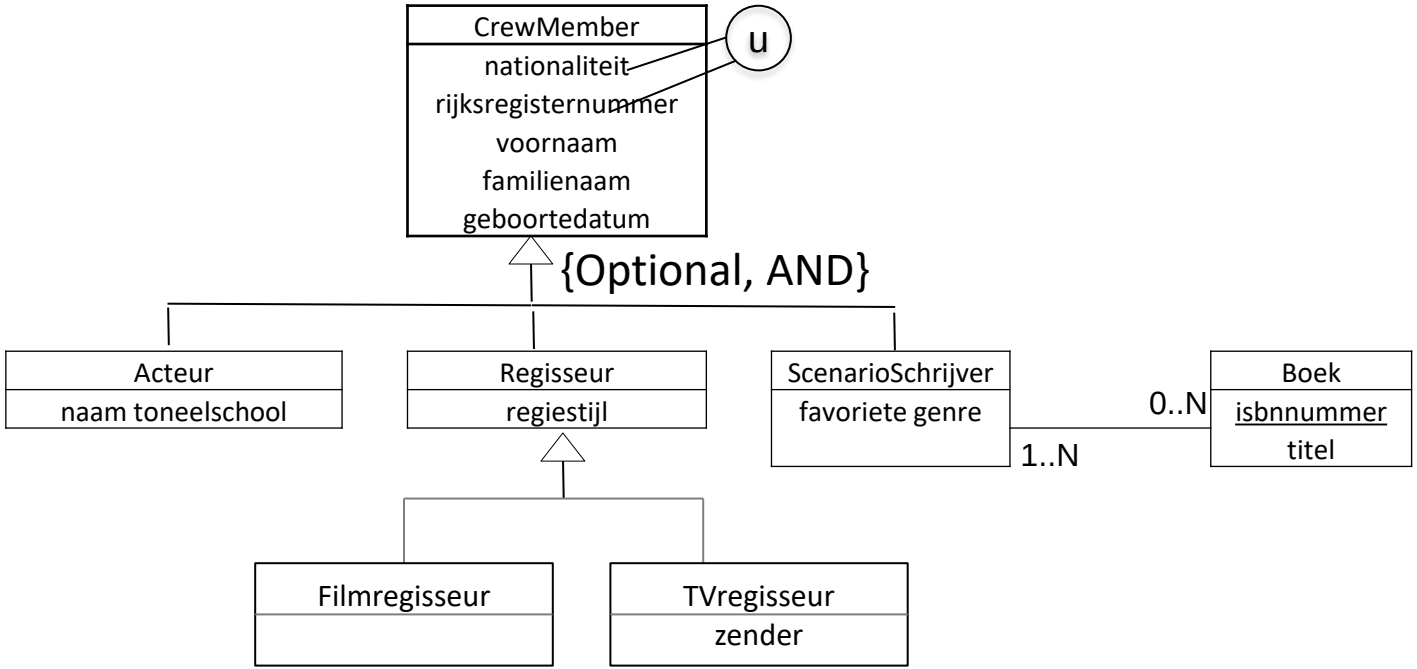
HO GENT

Specialisatie - Voorbeeld



HO GENT

Specialisatie - Voorbeeld



HO GENT

Specialisatie

- Om een subcollectie van entiteiten te karakteriseren, worden aparte entiteittypes aangemaakt: **sub(entiteit)types.**
- Het oorspronkelijke entiteittype is **supertype** voor deze subtypes.
- Een subtype erft alle attribuuttypes en relatietypes van zijn supertype.
- Een subtype kan:
  - zelf specifieke attribuuttypes hebben. Deze gelden niet voor het supertype.
  - aanleiding geven tot extra relatietypes met (andere)(sub) entiteittypes. Deze gelden niet voor het supertype.
- Een subtype heeft GEEN kandidaatsleutelattribuuttype!
- **Specialisatie** = het creëren van specifiekere subtypes voor een gegeven entiteittype.
  Een specialisatie definieert een **IS-EEN** relatie.

HO GENT

Generalisatie

- Generalisatie = het creëren van een algemeen supertype dat de gemeenschappelijke attribuuttypes en relatietypes van een aantal gegeven entiteittypes verenigt. Dit is het omgekeerde proces van specialisatie.

- Specialisatie komt overeen met een top-down proces van conceptuele verfijning.

- Generalisatie komt overeen met een bottom-up proces van conceptuele synthese.

HO GENT

Participatie en Disjoint constraint

- Participatie constraint
  Niet elk CrewMember moet op elk tijdstip Acteur, Regisseur of ScenarioSchrijver zijn.
  Er kunnen ook andere CrewMembers zijn. M.a.w. er kunnen entiteiten van het supertype zijn die niet tot één van de opgesomde subtypes behoren.
  => Optional

- Disjoint constraint
  Een CrewMember kan zowel Regisseur als ScenarioSchrijver, of Regisseur en Acteur zijn. M.a.w. een entiteit van het supertype kan tegelijkertijd voorkomen in meerdere subtypes.
  => And

HO GENT

Participatie constraint

- De participatierestrictie bepaalt of op elk tijdstip elke entiteit van het supertype ook entiteit *moet* zijn van ten minste één subtype of niet.

- Totale participatie
  - Elke entiteit van het supertype moet op elk tijdstip ook entiteit zijn van ten minste één subtype. Er bestaan geen entiteiten van het supertype die niet tot een subtype behoren.
    ⇨Mandatory

- Partiële participatie
  - Er kunnen entiteiten van het supertype zijn die niet tot één van de opgesomde subtypes behoren.
    ⇨ Optional

HO GENT

Disjoint constraint

- De Disjoint constraint bepaalt of een entiteit van het supertype tegelijkertijd kan voorkomen in meerdere subtypes of niet.

- Overlappende subtypes
  - Een entiteit kan tot meer dan één subtype behoren.
    ⇨ AND

- Disjuncte subtypes
  - Een exclusieve OR tussen de subtypes: een entitiiteit kan maar tot 1 subtype behoren.
    ⇨ OR

HO GENT

3. Oefeningen

HO
GENT

Participatie en Disjoint constraint

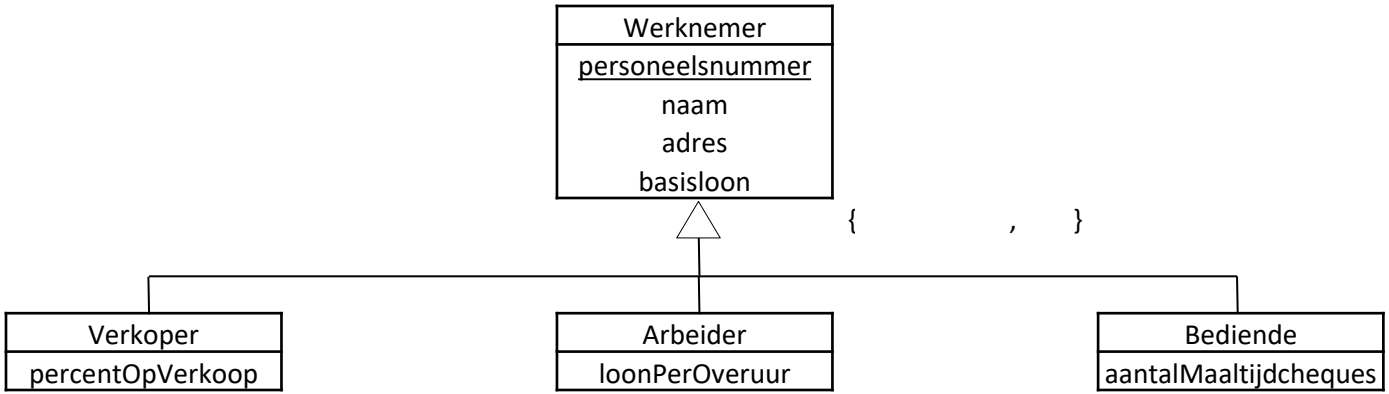
- Een werknemer in een meubelzaak kan één van de volgende functies hebben:
  - een verkoper
  - een arbeider
  - een bediende

- Alle 3 types werknemers hebben algemene kenmerken: personeelsNummer, naam, adres, basisloon.
- Er zijn ook specifieke kenmerken per soort:
  - een verkoper krijgt naast zijn basisloon ook een % op zijn omzetcijfer
  - een arbeider krijgt overuren uitbetaald
  - een bediende krijgt een aantal maaltijdcheques

HO GENT

Participatie en Disjoint constraint

- Participatie constraint → Mandatory of Optional?
- Disjoint constraint → And of Or?



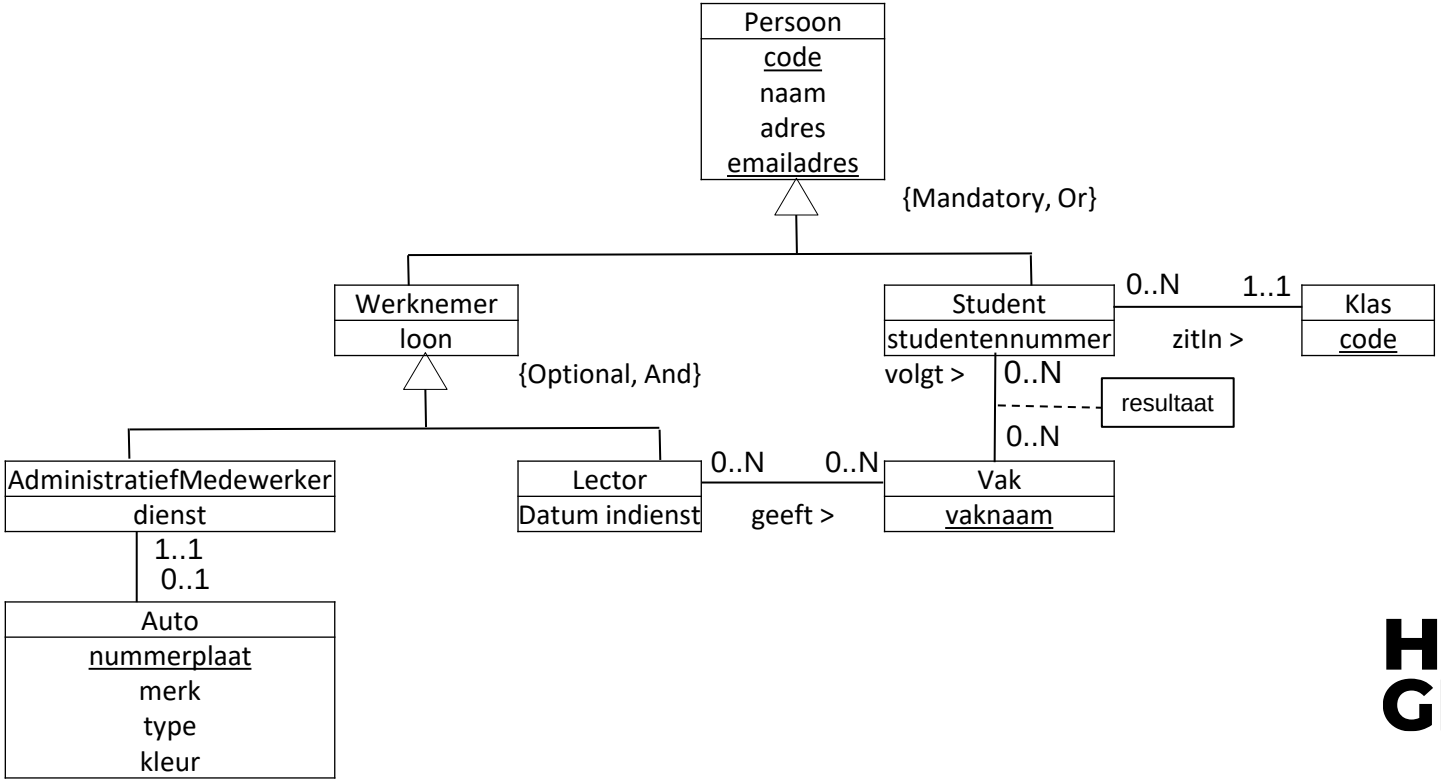
Oefening Hogeschool

Teken het EERD uit voor onderstaande situatie

- Een persoon in de DB van een hogeschool is werknemer of student. Elke persoon heeft een unieke code, een naam, een adres en een uniek e-mailadres.
- Het kan voorvallen dat een werknemer in deze hogeschool zowel administratieve taken uitvoert als doceert. Behalve administratieve medewerkers en lectoren zijn er nog andere werknemers in deze hogeschool. Er zijn geen jobstudenten.
- Een student zit in één welbepaalde klas (unieke code), een lector doceert een aantal vakken, studenten hebben een bepaald eindcijfer per vak.
- Sommige administratieve medewerkers hebben een firmawagen (unieke nummerplaat, merk, type, kleur).
  Een auto is altijd van een administratief medewerker.
- Van een student kennen we zijn studentenummer en van een lector de datum van indiensttreding.
- Een vak heeft een unieke vaknaam.
- Van een administratief medewerker weten we in welke dienst hij is tewerkgesteld.

HO GENT

Oefening Hogeschool



4. Beperkingen van het ERD

HO
GENT

Beperkingen van het ERD - 1

- **Tijdelijke beperkingen (dit zijn beperkingen die gelden in een bepaald tijdsinterval) kunnen niet worden gemodelleerd.**

- Bijvoorbeeld
  - een project moet binnen een maand toegewezen worden aan een specifiek departement
  - een werknemer kan niet terugkeren naar een departement waarvan hij vroeger ooit de manager was
- Oplossing
  - Deze beperkingen moeten worden gedocumenteerd en kunnen later worden geïmplementeerd (bijvoorbeeld door databanktriggers - zie Databanken 2)

Beperkingen van het ERD - 2

- **Het ERD kan geen consistentie garanderen tussen verschillende relatietypes**

- Bijvoorbeeld
  - een werknemer moet werken in het departement waarvan hij de manager is
  - een werknemer kan enkel werken aan projecten die toegewezen zijn aan het departement waar hij werkt
- Oplossing
  - Deze beperkingen moeten worden gedocumenteerd en kunnen later worden geïmplementeerd (bijvoorbeeld door databanktriggers - zie Databanken 2)

HO GENT

Beperkingen van het ERD - 3

- **In het ERD is het domein waartoe attributen behoren, niet bekend. Men kan niet de mogelijke verzameling van waarden voor een attribuut vastleggen.**

- Bijvoorbeeld
  - het aantal uren dat aan een project gewerkt wordt, moet groter of gelijk zijn dan 0
- Oplossing
  - Deze beperkingen moeten worden gedocumenteerd en kunnen later worden geïmplementeerd (bijvoorbeeld door databankconstraints - zie Databanken 2)

HO GENT

Beperkingen van het ERD - 4

- **In het ERD is het niet mogelijk de definitie van functies op te nemen.**

- Bijvoorbeeld
  - de manier waarop het salaris van een werknemer wordt berekend

HO GENT

5. oefeningen

HO
GENT

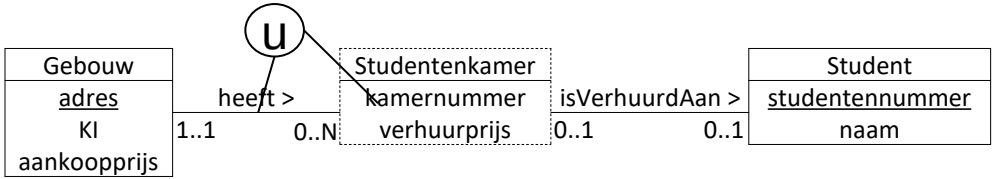
Wanneer is het ERD correct?

- Als het alle vragen naar informatie van de gebruiker kan beantwoorden.

HO
GENT

Wanneer is het ERD correct?

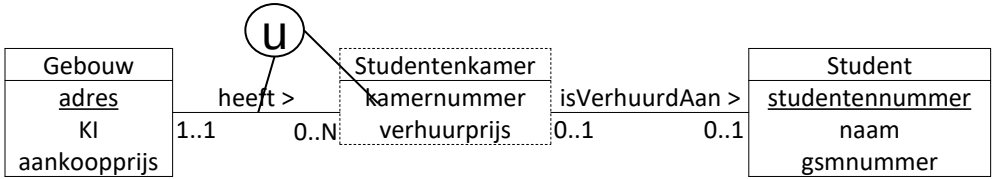
- Het model moet in staat zijn om volgende vragen te beantwoorden:
  - Aan welke student (naam) is een bepaalde studentenkamer momenteel verhuurd?
  - Wat is de verhuurprijs van een bepaalde studentenkamer?
  - Wat is het gsm-nummer van de student die kamer nummer 5 uit de Kerkstraat 12 in Gent huurt?



HO GENT

Wanneer is het ERD correct?

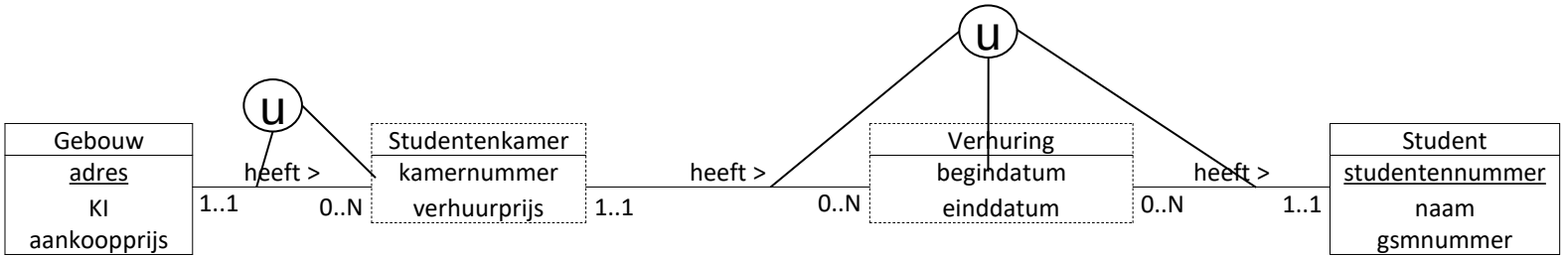
- Het model moet in staat zijn om volgende vragen te beantwoorden:
  - Aan welke student was een bepaalde studentenkamer vorig jaar verhuurd?



HO GENT

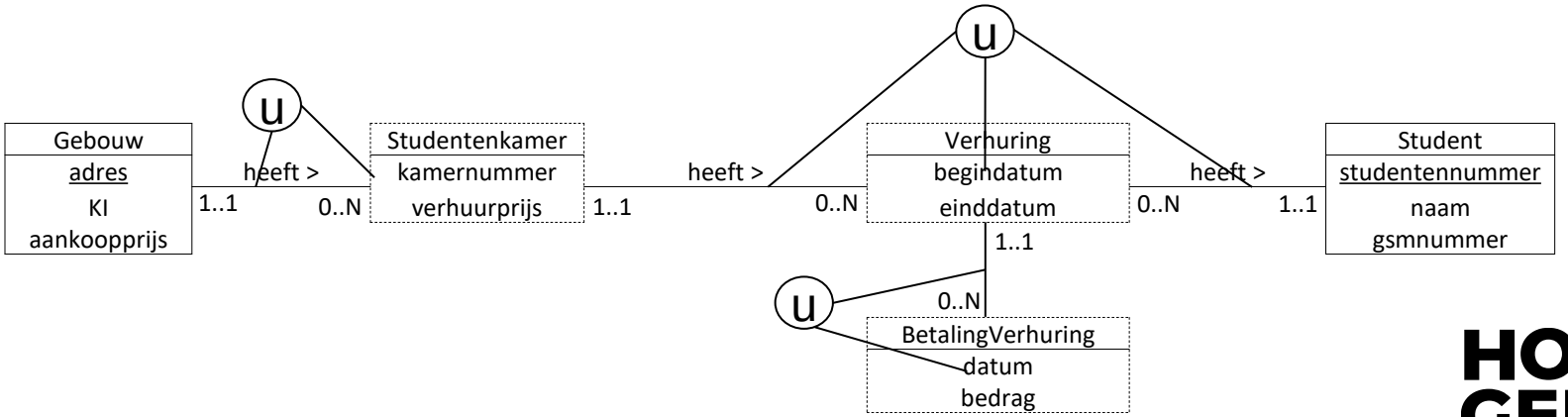
Wanneer is het ERD correct?

- Het model moet in staat zijn om volgende vragen te beantwoorden:
  - Heeft de huurder (de student) deze maand de huursom al betaald?



HO GENT

Wanneer is het ERD correct?



HO GENT

Oefening Blue-bike

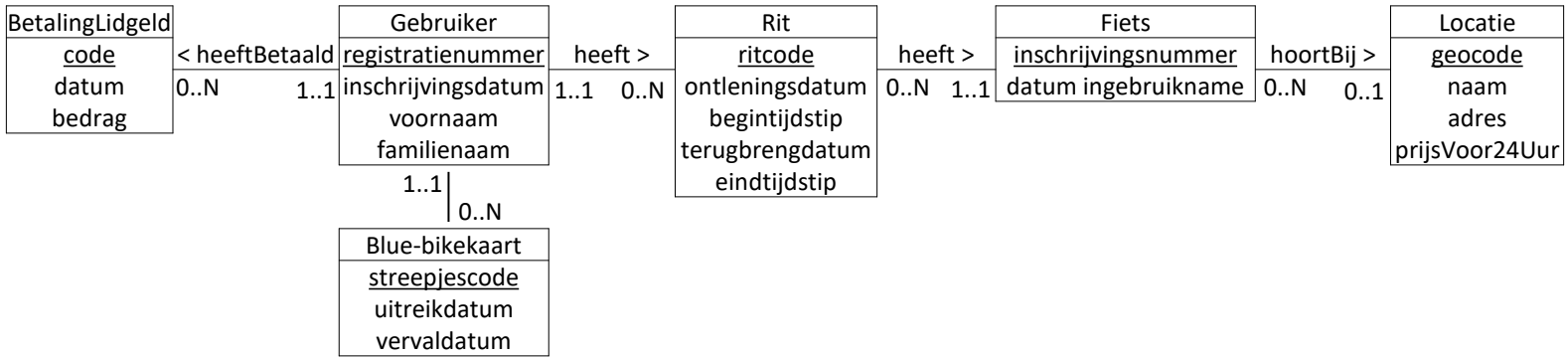
- Bluebike fietsen zijn deelfietsen die je kan ontlenen op meerdere locaties in België, vaak (maar niet altijd) aan treinstations. Zo kan je Blue-bike fietsen ontlenen aan het station én aan het stadhuis van Deinze.
- Je betaalt bij registratie en daarna jaarlijks € 12,5 voor je lidmaatschap.
- Daarvoor krijg je een Blue-bikekaart met unieke streepjescode waarmee je een fiets kan ontlenen op een locatie. (Je legt de blue-bikekaart op de kaartlezer van de blue-bikeautomaat en maakt de blue-bike los met de vrijgegeven sleutel.)
- Stel dat je een Blue-bikekaart verliest, dan krijg je een nieuwe kaart, maar de oude kaart blijft geregistreerd in de databank.
- Je moet de blue-bike steeds terugbrengen naar die locatie waar je hem ontleende en stopt de sleutel daarna opnieuw in de automaat.
- Het rittarief varieert afhankelijk van de locatie tussen de € 0, € 1,15 of € 3,15 per rit van 24 uur. Op enkele uitzonderingen na betaal je bijna overal € 1,15 (24 uur).

HO GENT

Oefening Blue-bike

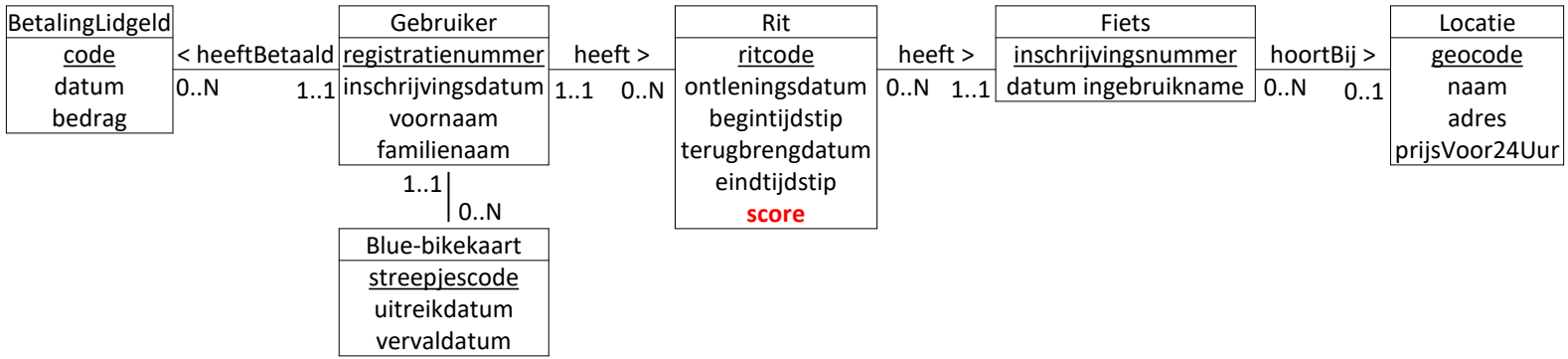
Pas het EERD aan

- Een gebruiker kan bij het inleveren van de sleutel in de automaat een score geven, over hoe hij de fietsrit ervaren heeft.



Oefening Blue-bike

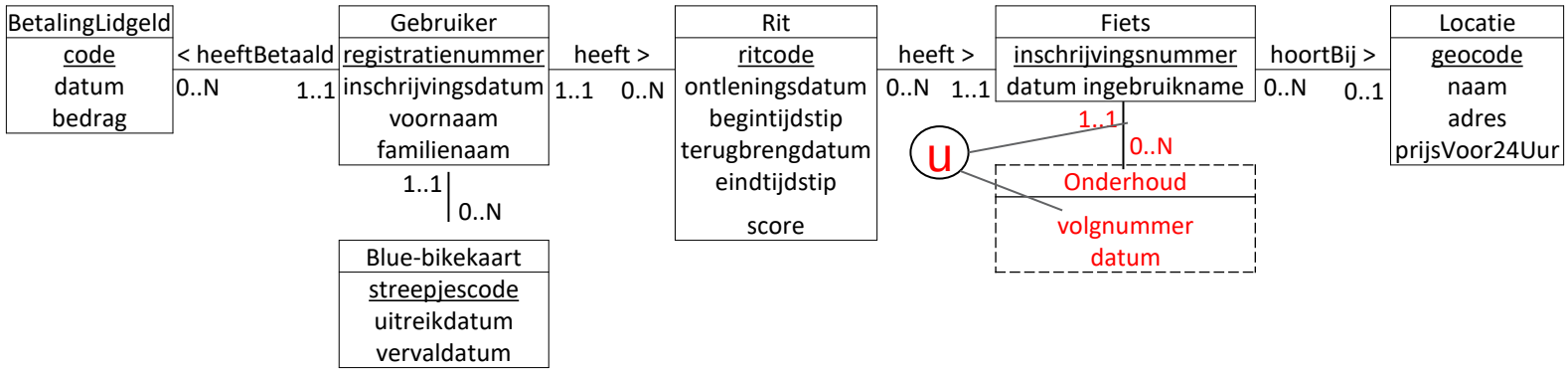
- Elke fiets krijgt regelmatig een onderhoud. Een onderhoud wordt genummerd per fiets en heeft een datum waarop het onderhoud werd uitgevoerd.



HO GENT

Oefening Blue-bike

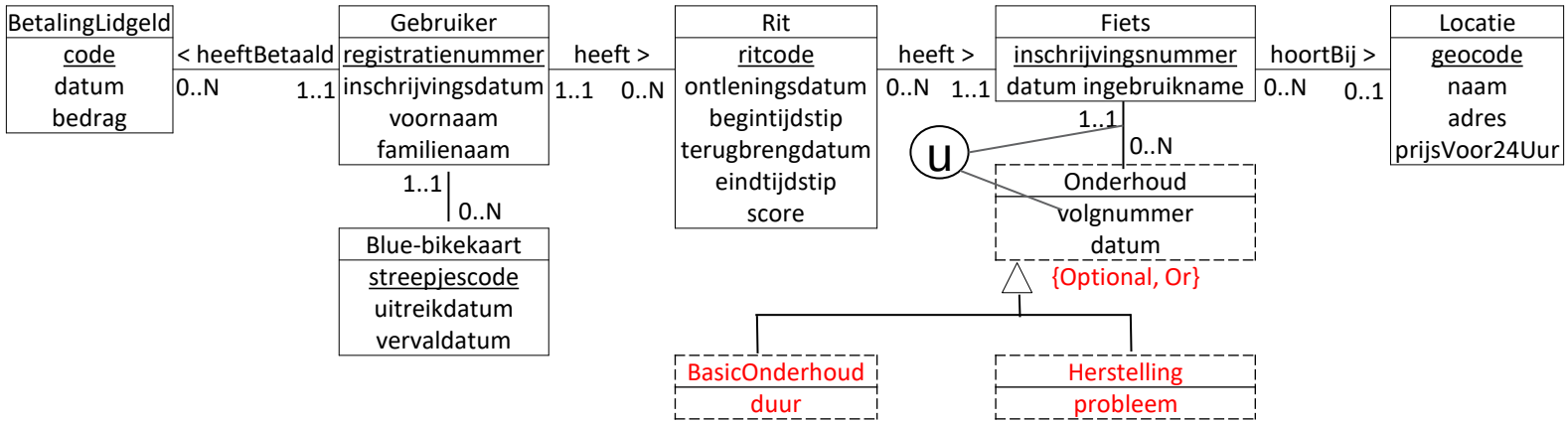
- Er zijn verschillende soorten onderhoud. Enerzijds is er een basic onderhoud waarbij de banden opgepompt worden, de ketting gesmeerd wordt, … Hiervan wordt de duur bijgehouden. Anderzijds is er een herstelling. Bij een herstelling wordt een korte omschrijving van het probleem bijgehouden. Er kunnen ook nog andere soorten Onderhoud zijn.



HO GENT

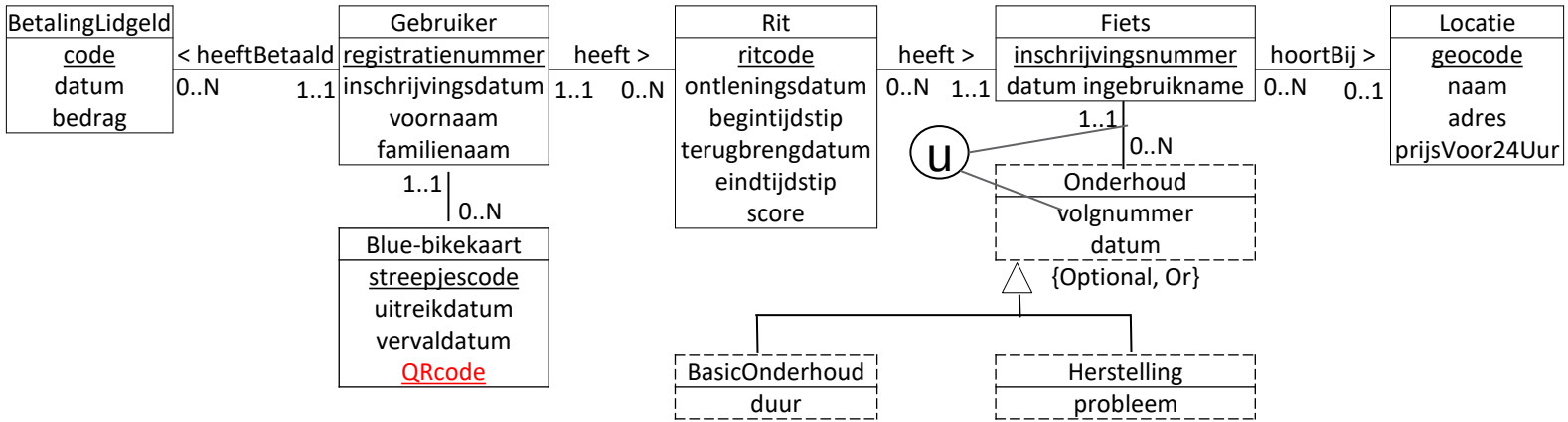
Oefening Blue-bike

- Een Blue-bikekaart kan niet enkel een streepjescode maar ook een unieke QR code hebben.



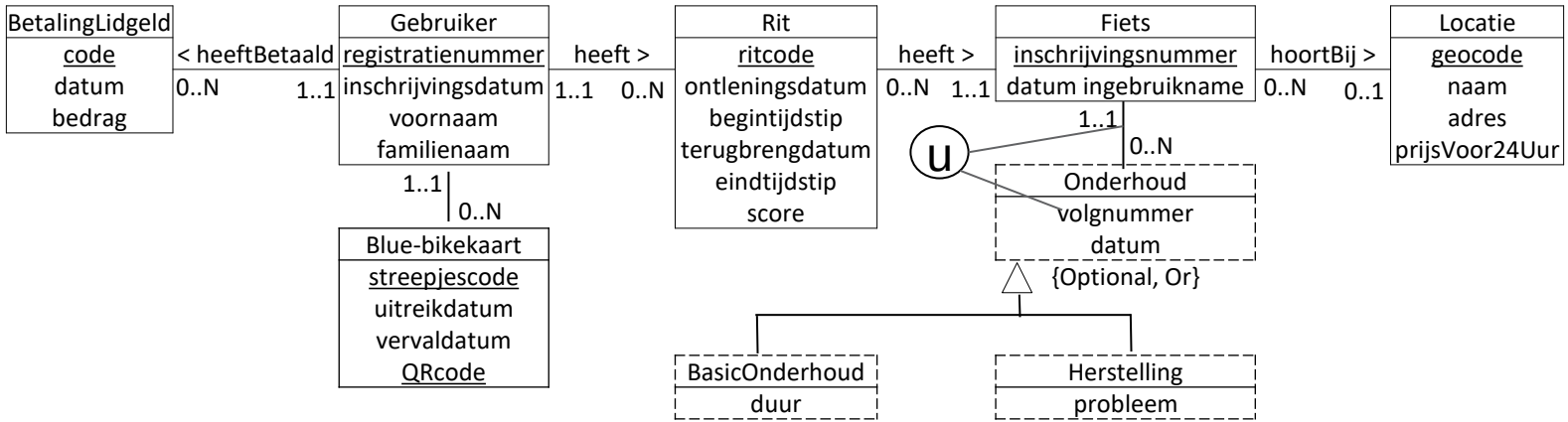
Oefening Blue-bike

- Het moet op elk ogenblik mogelijk zijn om het totaal aantal beschikbare fietsen per locatie op te vragen.



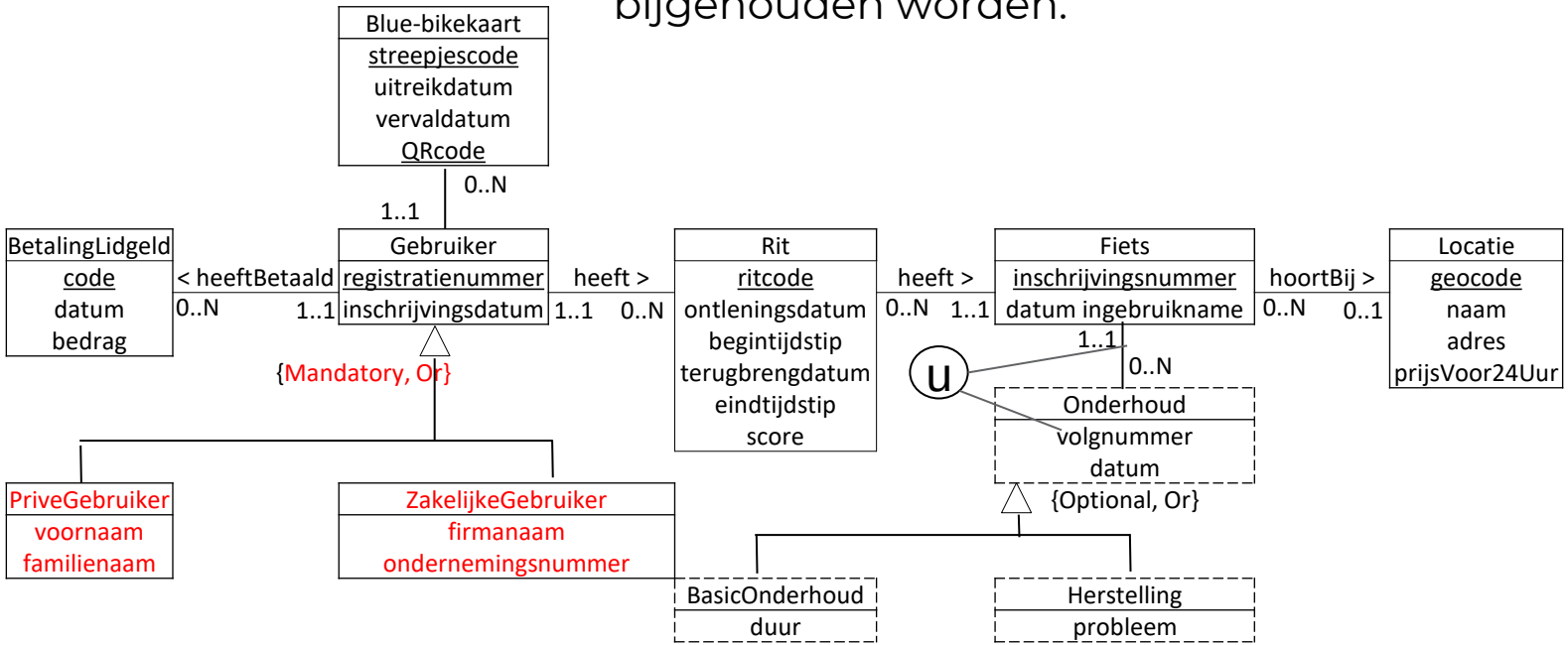
Oefening Blue-bike

- Om facturatieredenen moet er vanaf nu een onderscheid gemaakt worden tussen privé gebruikers (voornaam, familienaam) en zakelijke gebruikers (firmanaam, ondernemingsnummer).



Oefening Blue-bike

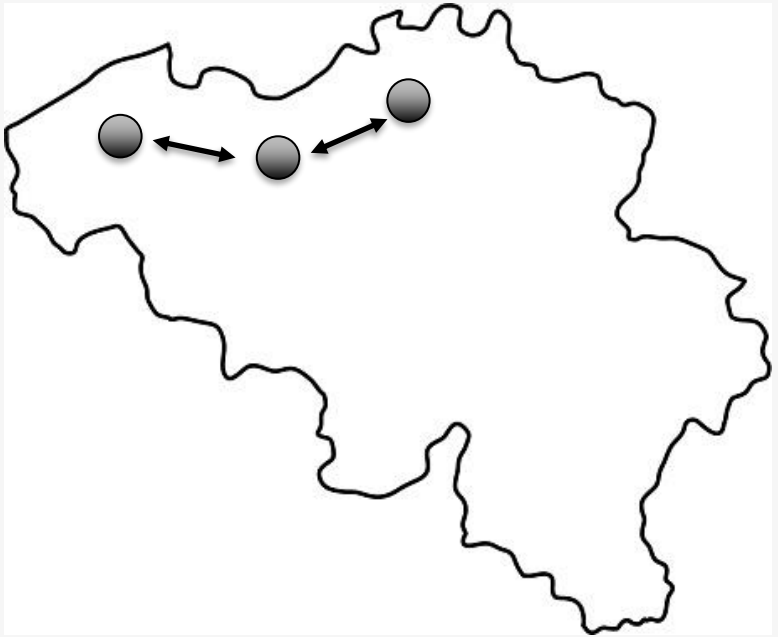
- Voor elke locatie willen we bijhouden welk de dichtstbijzijnde locatie (in vogelvlucht) is én moet de afstand tot deze dichtstbijzijnde locatie bijgehouden worden.

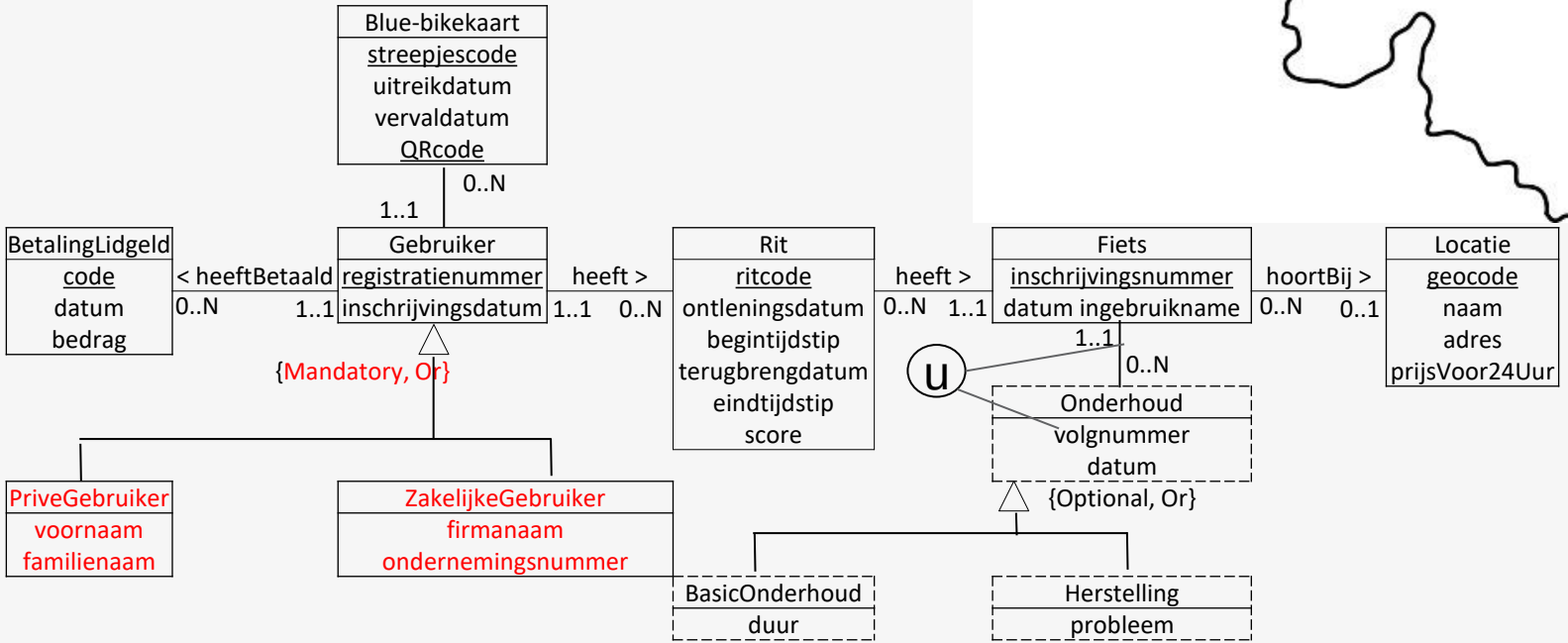


HO GENT

Oefening Blue-bike

- Dichtst bijzijnde locatie?
  Cardinaliteit?

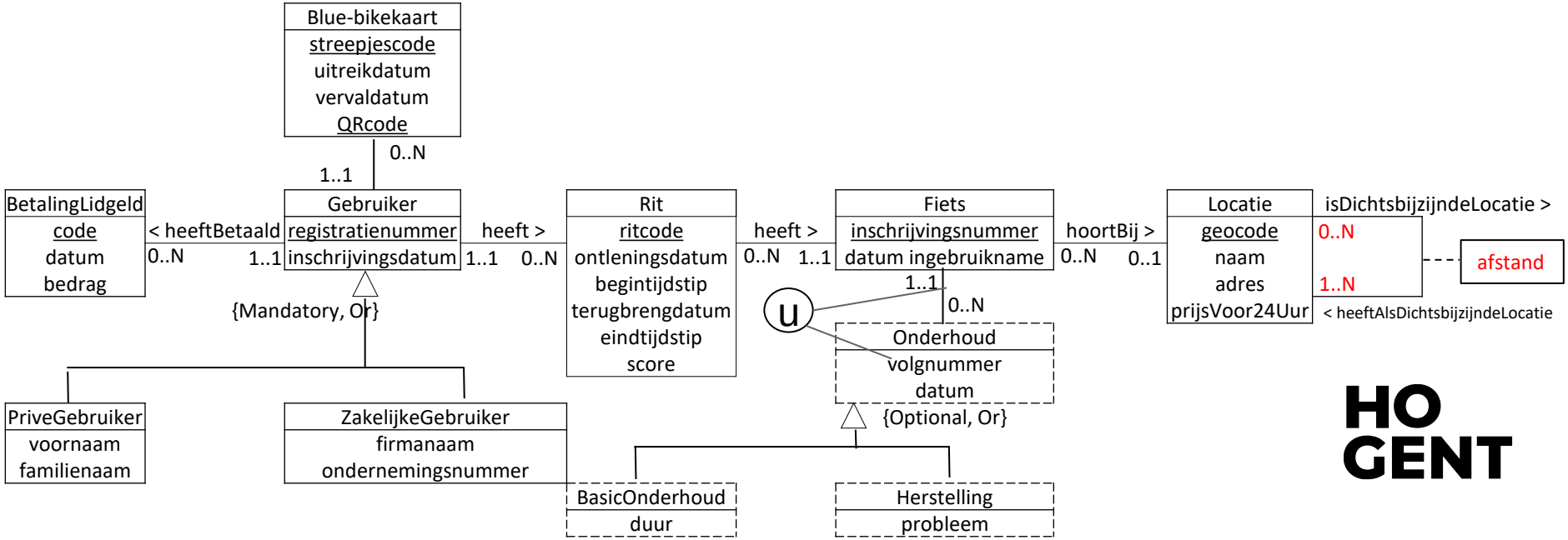




HO GENT

Oefening Blue-bike

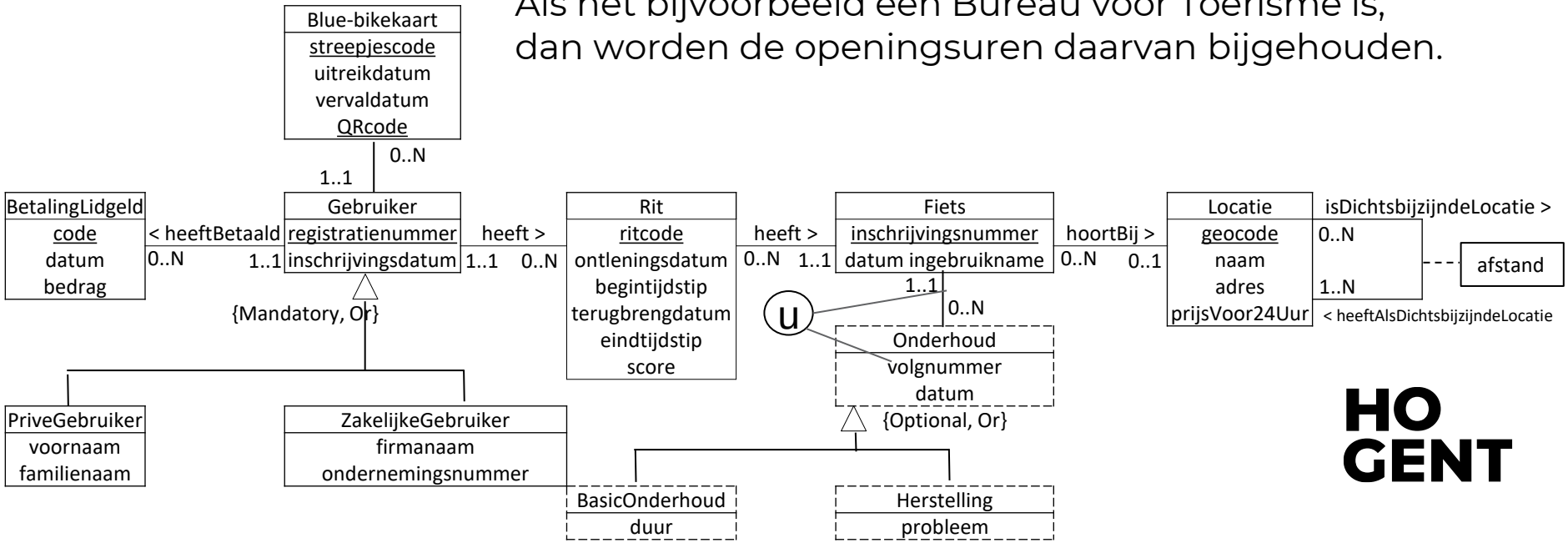
- Een gebruiker kan met één kaart 2 fietsen tegelijkertijd uitlenen: de gebruiker leent eerst 1 fiets uit en daarna nog een fiets (zonder meerkost).



HO GENT

Oefening Blue-bike

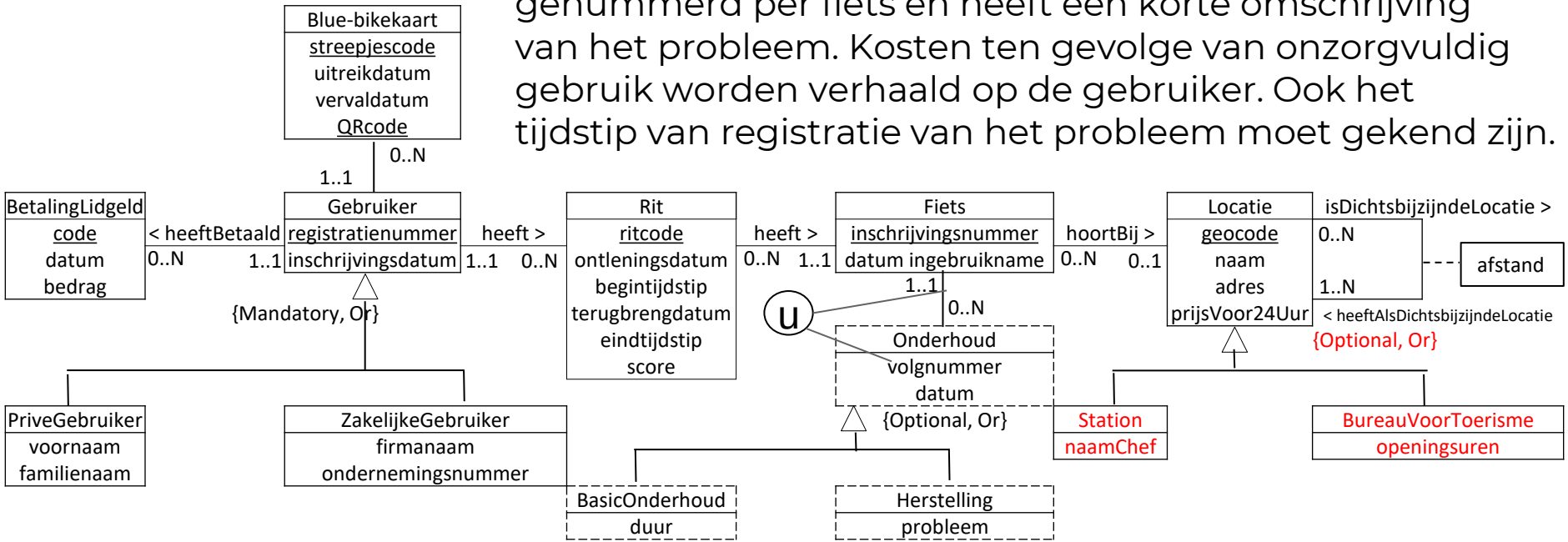
- Indien een Locatie een treinstation is, willen we extra informatie bijhouden over dat treinstation, bijvoorbeeld de naam van de stationschef.
  Als het bijvoorbeeld een Bureau voor Toerisme is, dan worden de openingsuren daarvan bijgehouden.



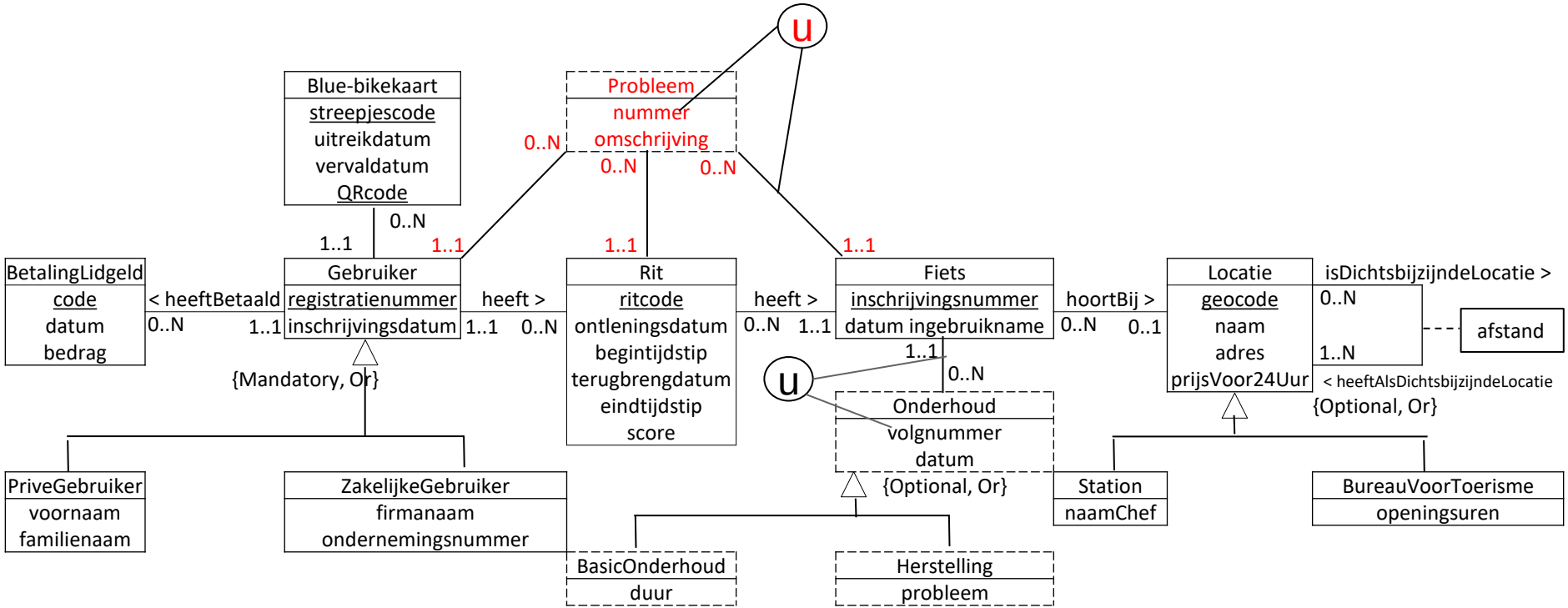
HO GENT

Oefening Blue-bike

- Een gebruiker kan na een rit eventuele problemen met de fiets onmiddellijk melden (voorlicht defect, linkerrem defect, …). Een probleem wordt genummerd per fiets en heeft een korte omschrijving van het probleem. Kosten ten gevolge van onzorgvuldig gebruik worden verhaald op de gebruiker. Ook het tijdstip van registratie van het probleem moet gekend zijn.



Oefening Blue-bike



Databases

H5 Relationeel model

HO
GENT

H5. Relationeel model

1. Inleiding
2. Specialisatie
3. Oefeningen
4. Beperkingen van het ERD
5. Oefeningen

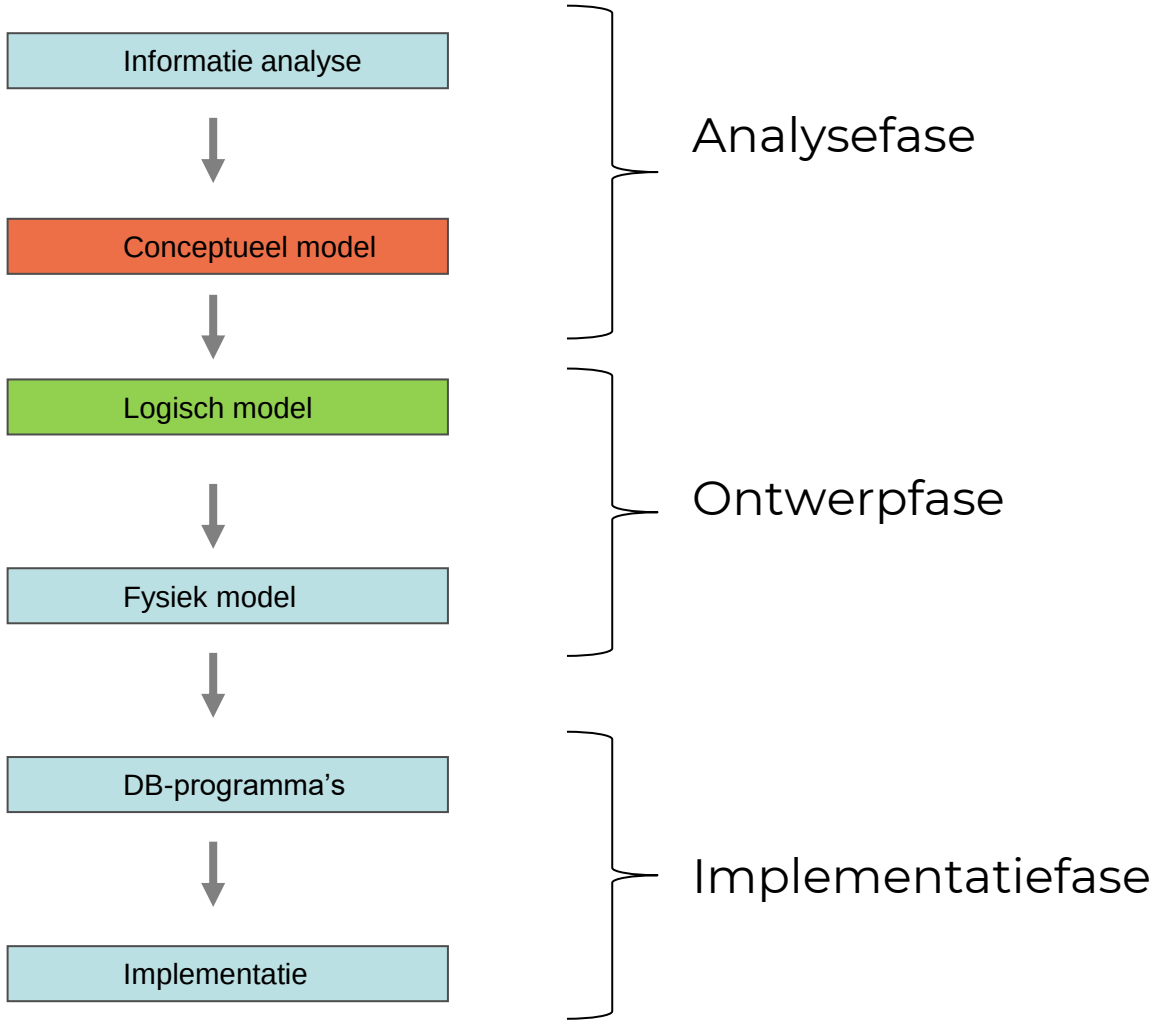
HO GENT

1. Inleiding

HO
GENT

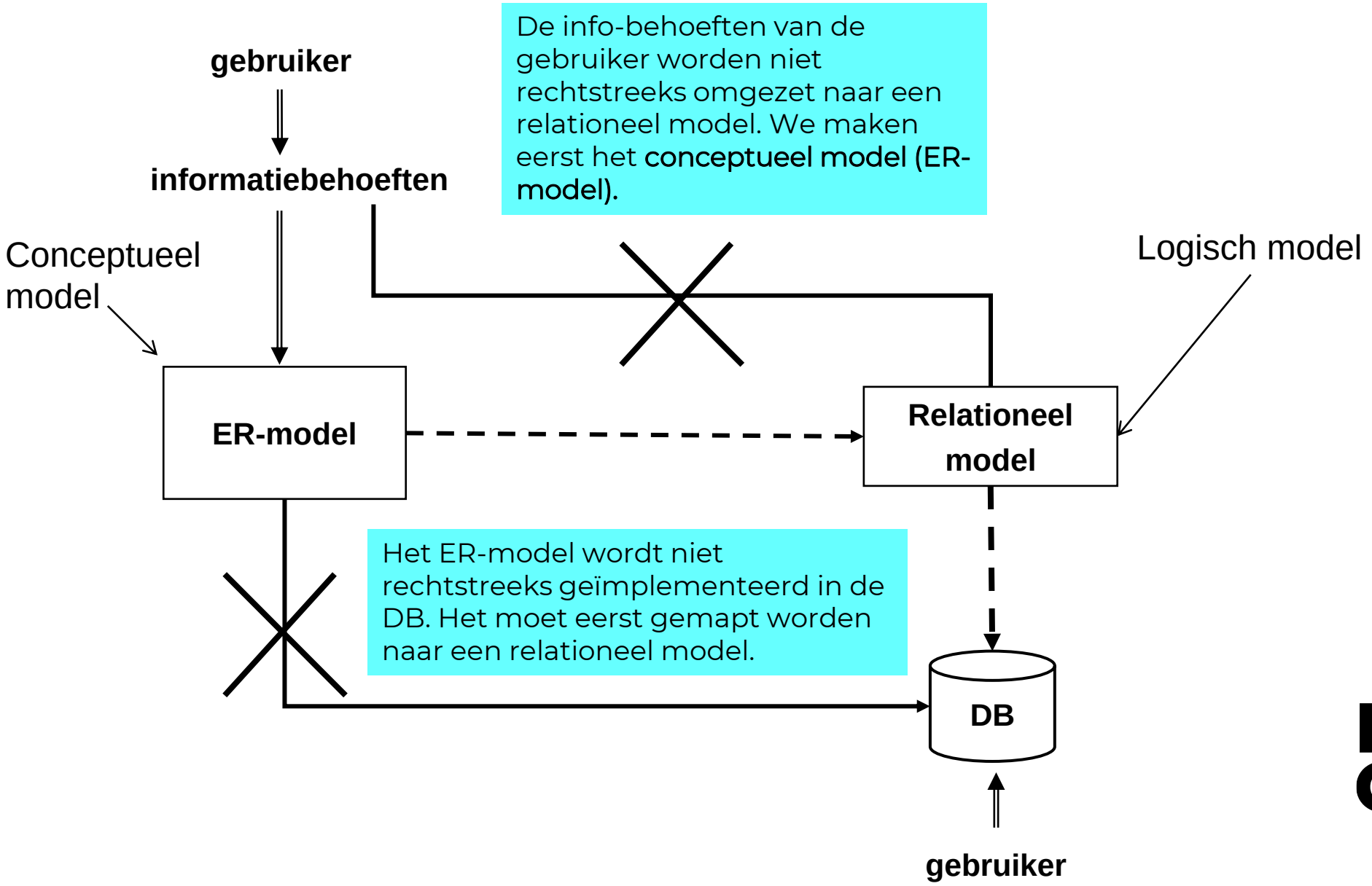
Fasen in de ontwikkeling van een DB

Stappen in de ontwikkeling van een databank:



HO GENT

Relationeel model



HO GENT

Relationeel model

Een relationeel model bestaat uit **tupelverzamelingen** die met elkaar verbonden zijn via **vreemde sleutels**.

Terminologie:

- tupel
- tupelverzamelingen
- attribuut
- domein
- attribuuttype
- sleutels

HO GENT

Definities

- Tupel (record)
  - Een **geordende** lijst met **waarden van kenmerken** die een object beschrijven.
  - Een tupel is **steeds uniek**.

- Voorbeeld:
  Een tupel voor het beschrijven van een auto:
  (kx57 754, rood, Aston Martin, 2)

HO GENT

Definities

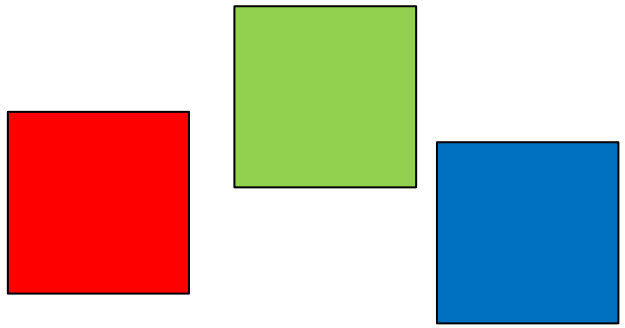
- Attribuut
  - Een **benoemd kenmerk** van een tupel.
  - Een attribuut van een tupel **mag geen meerdere waarden hebben**, de attribuutwaarde moet éénwaardig en atomair (ondeelbaar) zijn.
- Voorbeeld:
  Het attribuut "*rood*", een benoemd kenmerk van een auto.
  (kx57 754, **rood**, Aston Martin, 2)

HO GENT

Definities

- Domein
  Een **beperkte** verzameling van **mogelijke (toegelaten) waarden** die voor de attributen in de tupels van een relatie kunnen worden gebruikt.

- Voorbeeld:
  Het domein "*kleuren*" is een **beperkte** verzameling van alle mogelijke kleuren die een auto kan hebben: rood, groen, blauw, ….



HO GENT

Definities

- Attribuuttype (Datatype)
  Elk attribuut is van een bepaald type. Dit type wordt **afgeleid uit het domein**.

- Voorbeeld:
  Het attribuut 2 (waarde voor“*aantal deuren*”) is een aantal. Het **domein** zijn alle **getallen tussen 2 en 5**. Bijgevolg moet het **datatype numeriek** zijn.

HO GENT

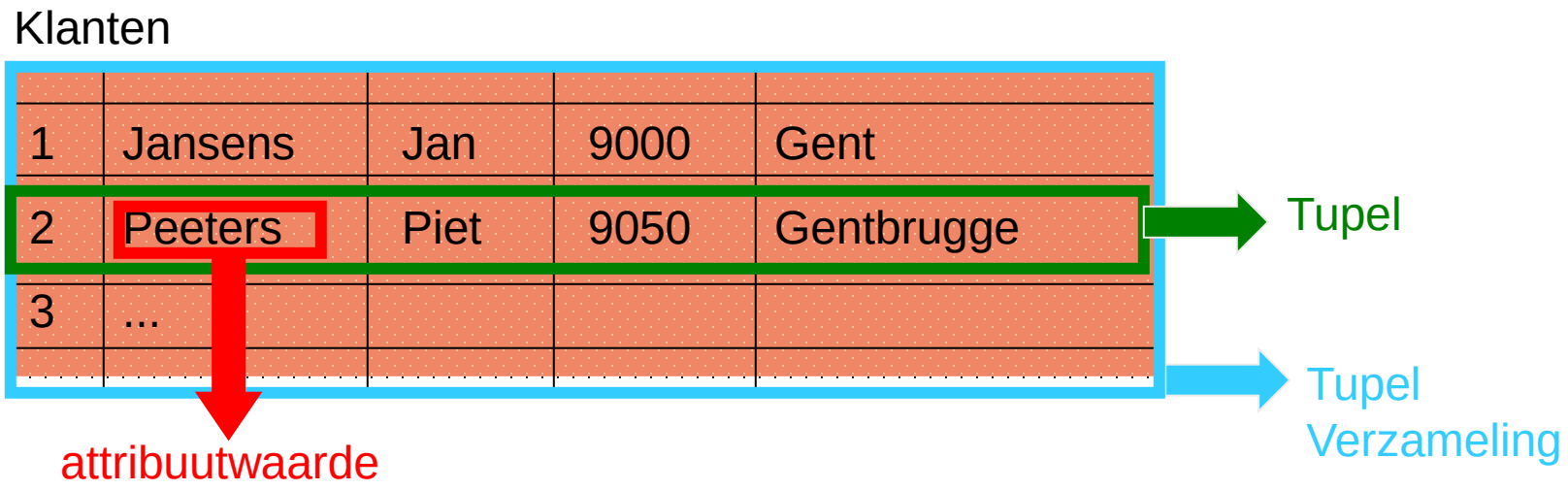
Definities

- Tupelverzameling
  Een verzameling van tupels die gelijksoortige objecten beschrijven.

- Voorbeeld:
  een verzameling van tupels die auto's beschrijven.
  {(kx57 754, rood, Aston Martin, 2), (xhd 352, groen, Porsche, 2), (123 klm, zwart, Maybach,2)}

HO GENT

Definities: samengevat



- Tupelverzameling ⇨ tabel
- Tupel = geordende lijst met attribuutwaarden ⇨ rij
  - ✓ In verzameling is elke tupel uniek.
- Attribuutwaarde = inhoud van 1 veld ⇨ cel
  - ✓ Moet éénwaardig en atomair zijn.

HO GENT

ER-model ◄== ► Relationeel model

ER Model	Relationeel model
Entiteittype	Tupel Verzameling (tabel)
Entiteiten	Tupel (record)
Attribuuttype	Attribuuttype
1-1; 1-N; N-N, 1ste graad, 2e graad	1-1; 1-N,1e graad en 2e graad
en >	(Geen hogere)

HO GENT

Regels

Elke tupelverzameling in het relationele model moet voldoen aan volgende regels:

- elke tupel is **uniek**
- elk attribuut is **éénwaardig.** (kan maar 1 waarde hebben)
- elk attribuut is **atomair** (kan niet meer opgesplitst worden)
- verbanden tussen relaties worden gelegd aan de hand van **vreemde sleutels**

HO GENT

Elk tupel is uniek

- Sleutels
  zorgen ervoor dat elk tupel uniek is.

- Welk attribuut maakt elk tupel in deze verzameling uniek?

{(kx57754, rood, Aston Martin, 2),
(xhd352, groen, Porsche, 2),
(123klm, zwart, Maybach, 2),
(456aze, rood, Maybach, 2)}

HO
GENT

Elk tupel is uniek

- Uniek attribuut: nummerplaat
  {
    (**kx57754**, rood, Aston Martin, 2),
    (**xhd352**, groen, Porsche, 2),
    (**123klm**, rood, Maybach, 2),
    (**456aze**, geel, Maybach, 2)
  }

⇨ nummerplaat = sleutel

HO
GENT

Elk tupel is uniek

Er zijn verschillende soorten sleutels:

- kandidaatsleutels
- primaire sleutel
- alternatieve sleutels

HO GENT

Sleutels

Kandidaatsleutel

- Is een **minimale verzameling** van attributen in de tupel, waarvan de combinatie elk tupel, binnen een tupelverzameling, **uniek** kan identificeren.

- Voorbeeld (enkelvoudig)
  - Een student binnen HOGENT wordt uniek gezien op basis van zijn **StudentNummer.**

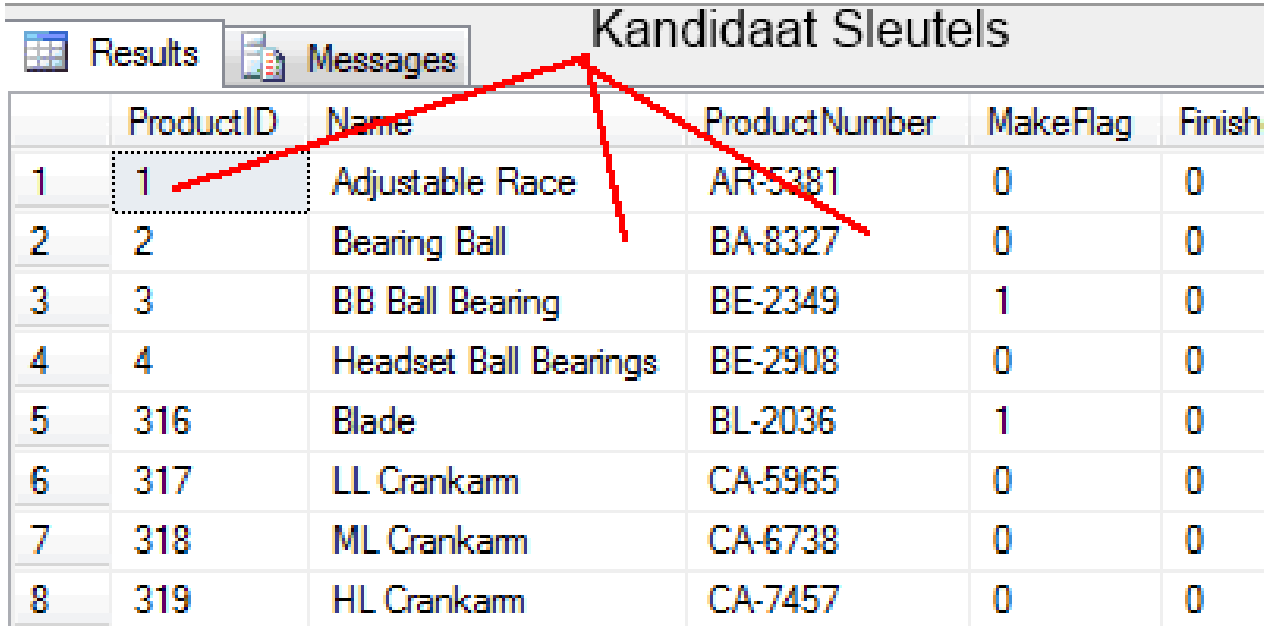
- Voorbeeld (samengesteld)
  - Het vluchtnummer en de dag van vertrek volstaan om elke vlucht uniek te identificeren, maar elke attribuutwaarde op zich volstaat niet. **Concreet**: vandaag vertrekken meerdere vluchten en vluchten met vluchtnummer SN3037 vertrekken elke dag. In dat geval vormen meerdere sleutelattribuuttypen de kandidaatsleutel van een entiteittype.

HO GENT

Sleutels

Kandidaatsleutel

Voorbeeld:



	ProductID	Name	ProductNumber	MakeFlag	Finish
1	1	Adjustable Race	AR-5381	0	0
2	2	Bearing Ball	BA-8327	0	0
3	3	BB Ball Bearing	BE-2349	1	0
4	4	Headset Ball Bearings	BE-2908	0	0
5	316	Blade	BL-2036	1	0
6	317	LL Crankarm	CA-5965	0	0
7	318	ML Crankarm	CA-6738	0	0
8	319	HL Crankarm	CA-7457	0	0

HO GENT

Sleutels

Primaire sleutel

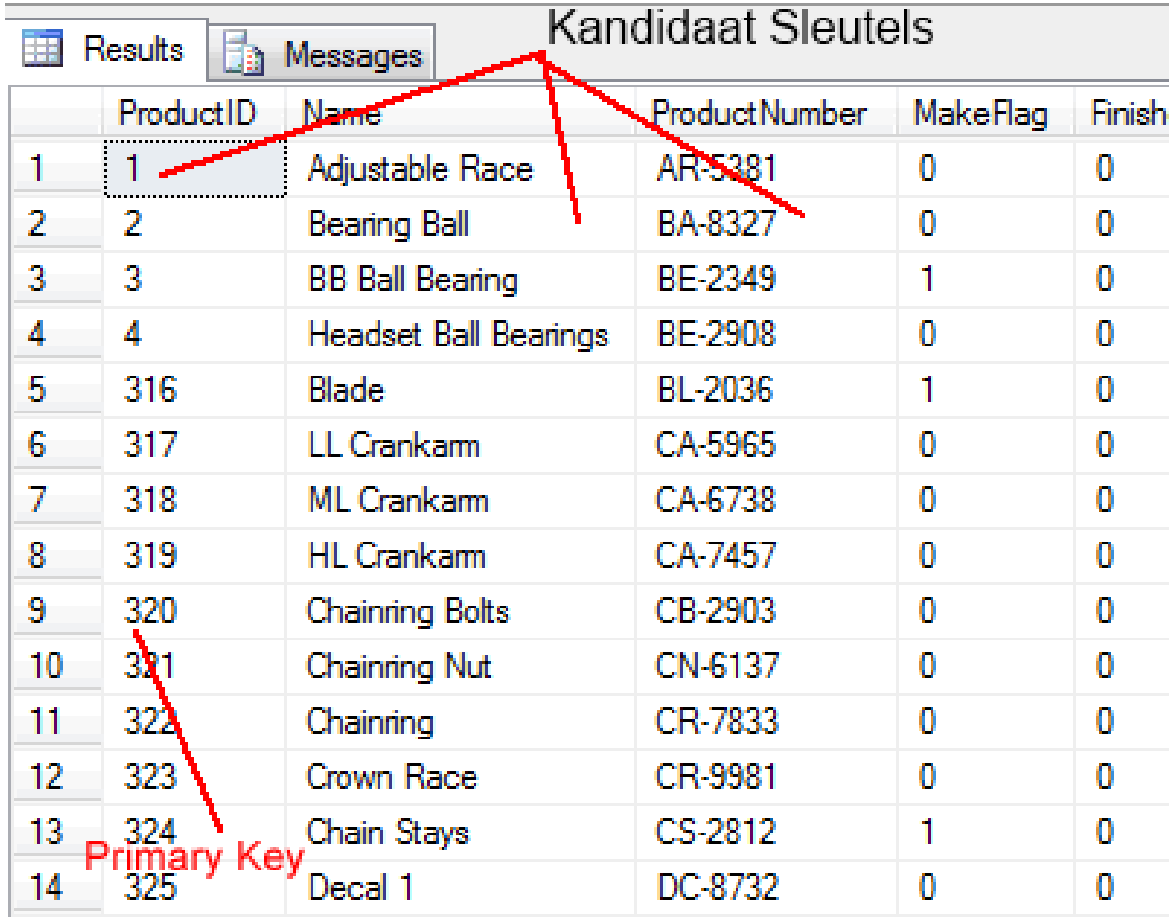
- Uit de kandidaatsleutels wordt één sleutel gekozen. Dit is de primaire sleutel.

- De primaire sleutel moet **steeds ingevuld** zijn (NULL niet toegelaten).

HO GENT

Sleutels

Primaire sleutel



	ProductID	Name	ProductNumber	MakeFlag	Finish
1	1	Adjustable Race	AR-5381	0	0
2	2	Bearing Ball	BA-8327	0	0
3	3	BB Ball Bearing	BE-2349	1	0
4	4	Headset Ball Bearings	BE-2908	0	0
5	316	Blade	BL-2036	1	0
6	317	LL Crankarm	CA-5965	0	0
7	318	ML Crankarm	CA-6738	0	0
8	319	HL Crankarm	CA-7457	0	0
9	320	Chainring Bolts	CB-2903	0	0
10	321	Chainring Nut	CN-6137	0	0
11	322	Chainring	CR-7833	0	0
12	323	Crown Race	CR-9981	0	0
13	324	Chain Stays	CS-2812	1	0
14	325	Decal 1	DC-8732	0	0

HO GENT

Sleutels

Alternatieve sleutel

- Elke **kandidaatsleutel die geen primaire sleutel geworden is**, wordt een **alternatieve sleutel** genoemd.

- **Opgelet**
  Indien in een tupel de alternatieve sleutel **NULL-waarde** heeft dan verliest de alternatieve sleutel zijn functie van ‘kandidaatsleutel’, want het is **niet langer uniek.**

HO GENT

Verbanden tussen relaties

Vreemde sleutel

HO
GENT

Verbanden tussen relaties

Vreemde sleutel

- Een vreemde sleutel (*foreign key*) is de **verbindende schakel** tussen twee tupelverzamelingen. Met een waarde uit een tupel van de ene verzameling kun je in een andere verzameling de juiste tupel met gerelateerde gegevens opzoeken.

- De vreemde sleutel in de ene tabel **verwijst naar de primaire sleutel** uit de andere tabel.

HO GENT

Verbanden tussen relaties

Vreemde sleutel - voorbeeld

Een project wordt steeds uitgevoerd in een bepaald departement.

Departement (<u>dnr</u>, dnaam, dlokatie, ...)

Project (<u>pnr</u>, pnaam, pduur, *departement*)

Het attribuut departement van Project is de vreemde sleutel die verwijst naar de primaire sleutel dnr van een Departement.

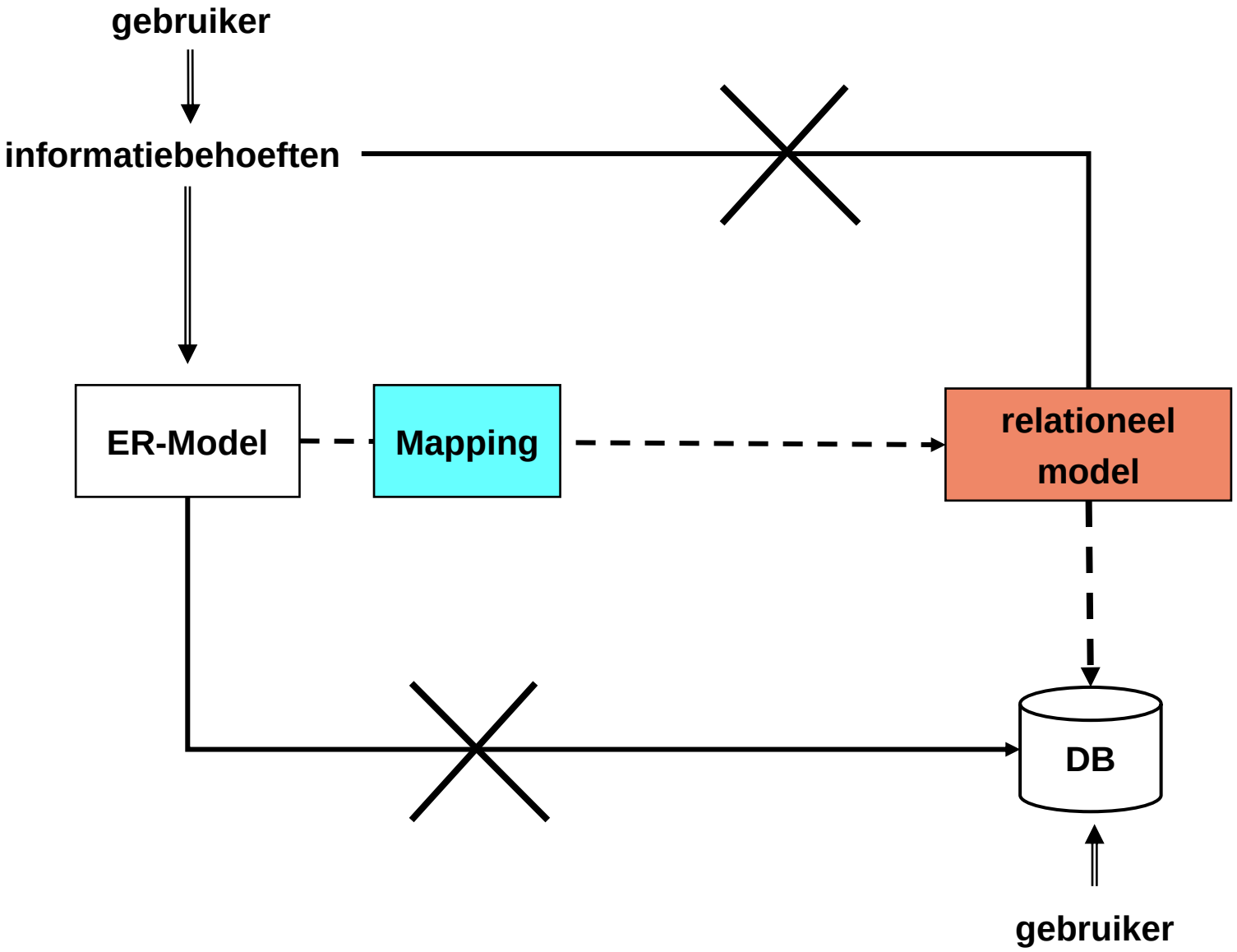
HO GENT

2. Mapping

HO
GENT

Mapping

= het omzetten van een ER-model (conceptueel model) naar een relationeel model (logisch model)



HO GENT

Mapping

Stappenplan

1. Elk entiteittype wordt een tupelverzameling of tabel (opgelet bij specialisatie!).
2. Enkelvoudige attribuuttypes overnemen.
3. Samengestelde attribuuttypes opsplitsen in enkelvoudige attribuuttypes.
4. Meerwaardige attributen in een aparte, nieuwe verzameling plaatsen.
5. Primaire sleutel bepalen (opgelet bij zwakke entiteiten!).
6. Voor elke relatie (verband) tussen entiteittypen de vreemde sleutel(s) bepalen.
7. Integriteitregels bepalen van elke vreemde sleutel.

HO
GENT

Mapping

Regels voor het bepalen van de vreemde sleutel

Verbanden tussen verzamelingen (tabellen) worden gelegd aan de hand van vreemde sleutels:

- **Binair** verband:
  - 1 op N verband: vreemde sleutel aan N-zijde
  - 1 op 1 verband: vreemde sleutel aan 1 zijde (zelf te kiezen)
  - Veel op veel: aparte tabel met 2 vreemde sleutels
- **Unair** verband:
  - 1 op veel: vreemde sleutel in zelfde (naam van rol 1-zijde!)
  - 1 op 1: vreemde sleutel met zichzelf
  - Veel op veel: nieuwe tabel met 2 vreemde sleutels (rolnamen gebruiken!)

HO GENT

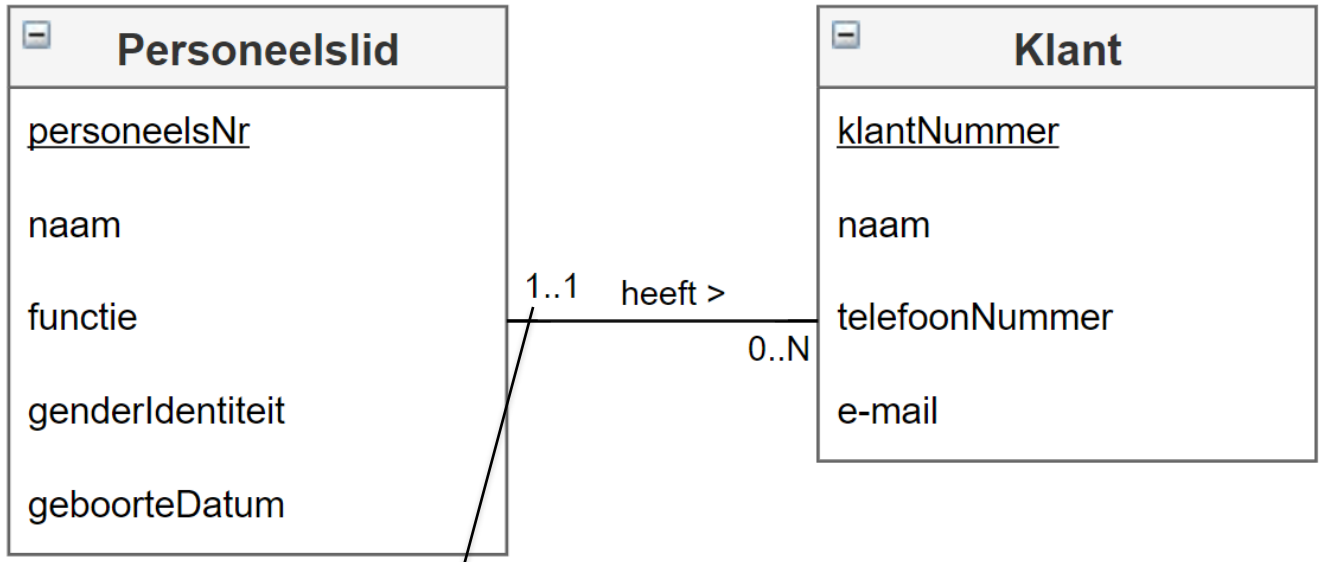
Mapping

Integriteitsregels vreemde sleutel:

- naar welke **primaire sleutel** verwijst deze vreemde sleutel
- is de vreemde sleutel **verplicht** of **optioneel** :
  dit bepaal je aan de hand van de minimumcardinaliteit
- is de vreemde sleutel **uniek** (bij 1 op 1)

HO
GENT

Mapping 1 op N binaire relatie



Vreemde sleutel in tabel aan N-zijde
⇨ verwijst naar 1 personeelslid

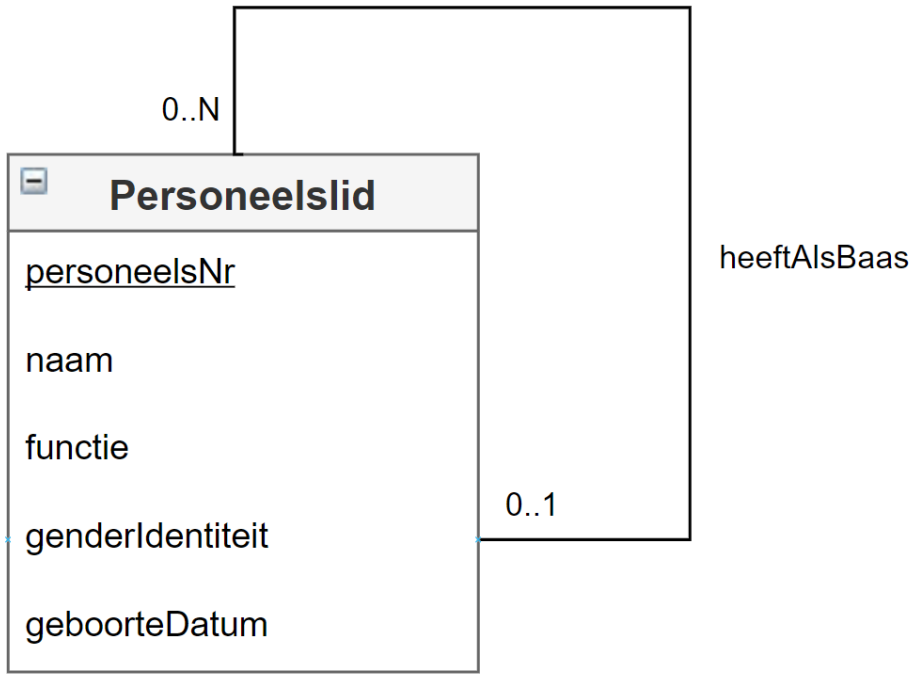
Min.card. = 1 ⇨ verplicht

Personeelslid (personeelsNr, voornaam, familienaam, functie, genderIdentiteit, geboortedatum)

Klant (klantNr, naam, telefoonNummer, eMail, personeelsNr)
IR: vreemde sleutel *personeelsNr* verwijst naar *personeelsNr* uit *Personeelslid*, is *verplicht*

HO GENT

Mapping 1 op N recursief (unaire relatie)



Rol 1-zijde opnemen als vreemde sleutel in zelfde tabel

- N-zijde: ondergeschikte: zijn er veel ⇨ indien je dit zou opnemen als vreemde sleutel dan zou dit veld meerwaardig zijn⇨ mag niet!
- 1-zijde: baas ⇨ een staflid heeft hoogstens 1 baas ⇨ dit opnemen als vreemde sleutel

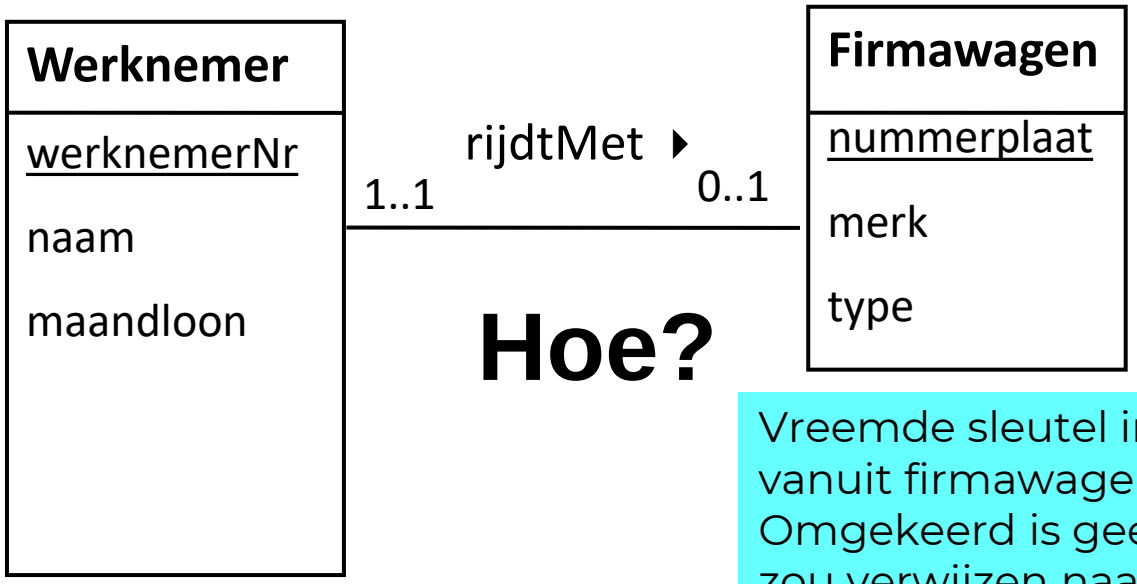
Min.card. = 0 ⇨ optioneel

Personeelslid (personeelsNr, voornaam, familienaam, functie, genderIdentiteit, geboortedatum, baas)

IR: VS baas verwijst naar personeelsNr, is optioneel

HO GENT

Mapping 1 op 1 relatie: verplichte deelname aan één zijde



Vreemde sleutel in tabel met optionele deelname:
vanuit firmawagen verwijzen naar werknemer
Omgekeerd is geen goed idee: indien men vanuit werknemer zou verwijzen naar firmawagen ⇨ veel lege velden aangezien niet elke werknemer een firmawagen heeft

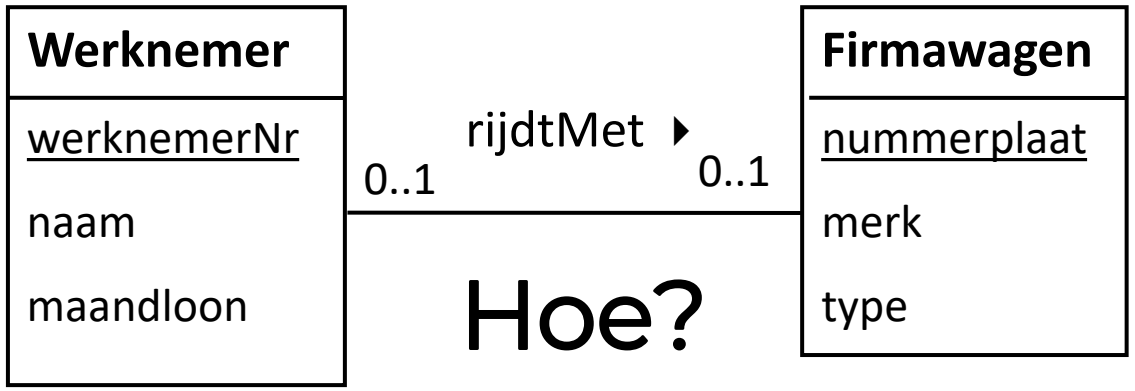
Werknemer (werknemerNr, voornaam, familienaam, maandloon)

Firmawagen (nummerplaat, merk, type, werknemer)
IR: Vreemde sleutel *werknemer* verwijst naar
werknemerNr uit *Werknemer* , is *verplicht,* ***uniek***

1..1 relatie ⇨ uniek

HO GENT

Mapping 1 op 1 relatie: optionele deelname aan beide zijden



Ofwel vanuit firmawagen verwijzen naar werknemer.

Ofwel vanuit werknemer verwijzen naar firmawagen.

Werknemer (werknemerNr, voornaam, familienaam, maandloon)

Firmawagen (nummerplaat, merk, type, werknemer)
IR: Vreemde sleutel *werknemer* verwijst naar *werknemerNr* uit *Werknemer, niet verplicht, uniek*
OF

Werknemer (werknemerNr, voornaam, familienaam, maandloon, auto)
IR: vreemde sleutel *auto,* verwijst naar *nummerplaat* uit *Firmawagen, optioneel, uniek*
Firmawagen (nummerplaat, merk, type)

HO GENT

Mapping 1 op 1 relatie: optionele deelname beide zijden

Beste oplossing?

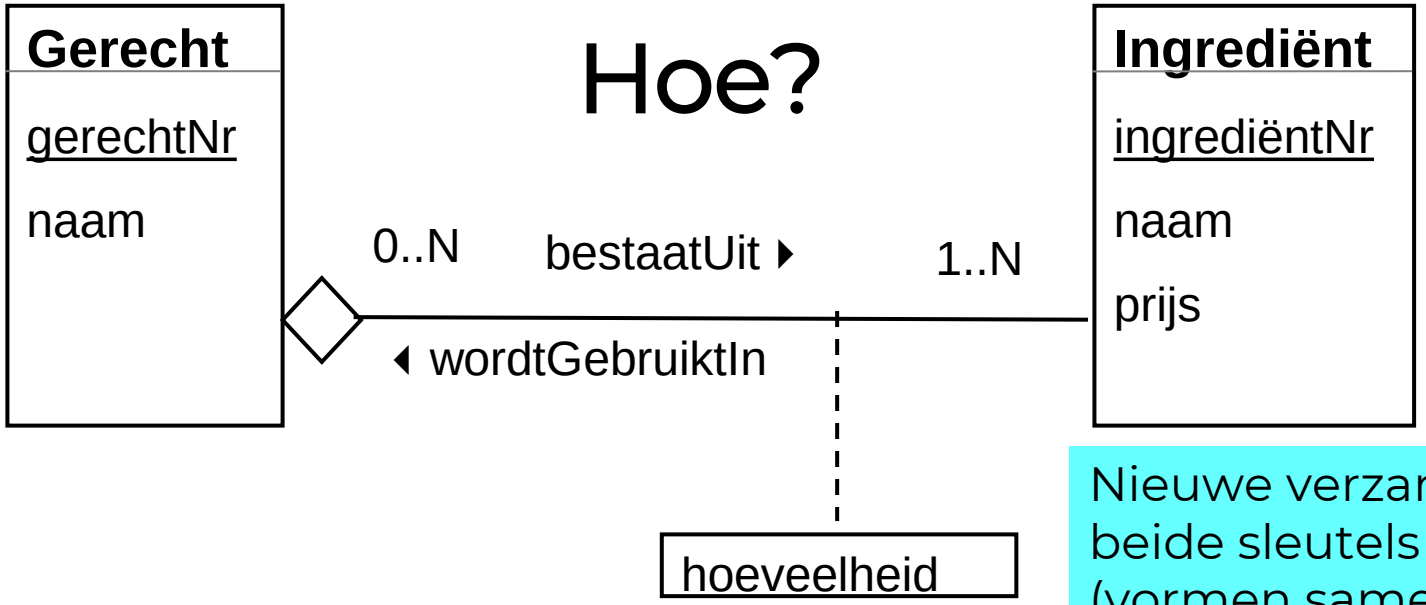
Werknemer (werknemerNr, voornaam, familienaam, maandloon)

Firmawagen (nummerplaat, merk, type, werknemer)
IR: Vreemde sleutel *werknemer* verwijst naar *werknemerNr* uit *Werknemer, niet verplicht, uniek*

Bovenstaande oplossing is de beste indien men aanneemt dat de meeste firmawagens aan een werknemer toebehoren en minder werknemers een firmawagen hebben.

HO GENT

Mapping veel op veel relatie



Nieuwe verzameling/tabel met beide sleutels als vreemde sleutel (vormen samen primaire sleutel in nieuwe relatie) + relatie-attributen.

Gerecht (gerechtNr, naam)

Ingrediënt (ingrediëntNr, naam, prijs)

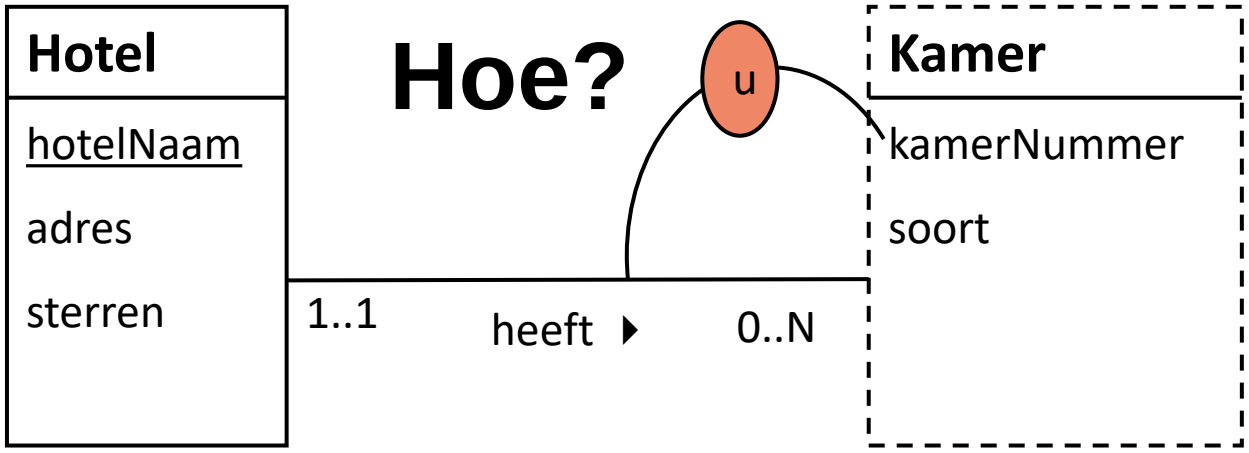
Ingrediënten/Gerecht (gerechtNr, ingrediëntNr, hoeveelheid)

- *gerechtNr:* vreemde sleutel, verwijst naar *gerechtNr* uit *Gerecht, verplicht*
- *ingrediëntNr:* vreemde sleutel, verwijst naar *ingrediëntNr* uit *Ingrediënt, verplicht*

Verplicht: want maakt deel uit van primaire sleutel!

HO GENT

Mapping zwak entiteittype



Kijken naar identificatie in ERD: hier wordt een kamer geïdentificeerd aan de hand van:

- zijn relatie met Hotel ➔ sleutel van Hotel = hotelnaam
- kamernummer

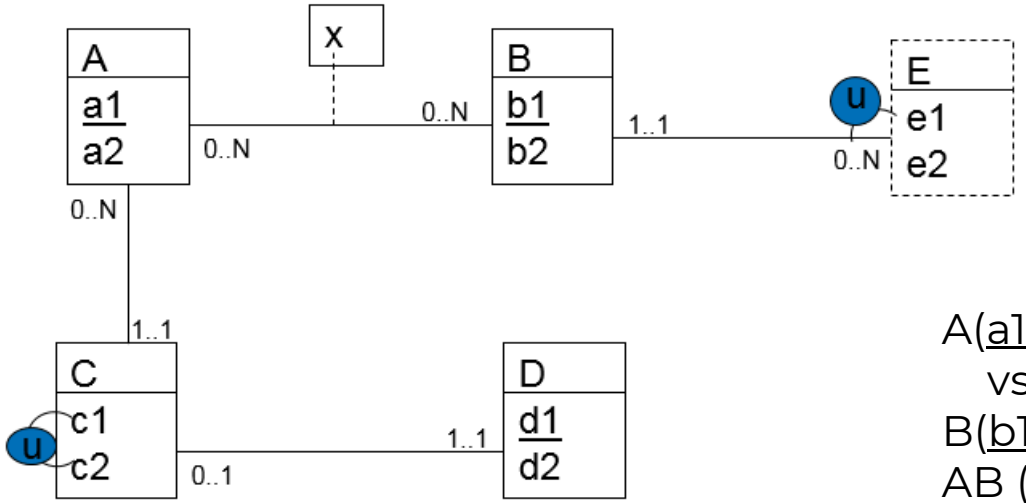
Niet vergeten om samengesteld attribuut adres op te splitsen (indien nodig)!!

Hotel (<u>hotelnaam</u>, straat, huisnummer, postcode, gemeente, land, sterren)

Kamer (<u>hotelnaam, kamernummer</u>, soort)
vreemde sleutel *hotelnaam,* verwijst naar *hotelnaam* uit *Hotel, verplicht*

HO GENT

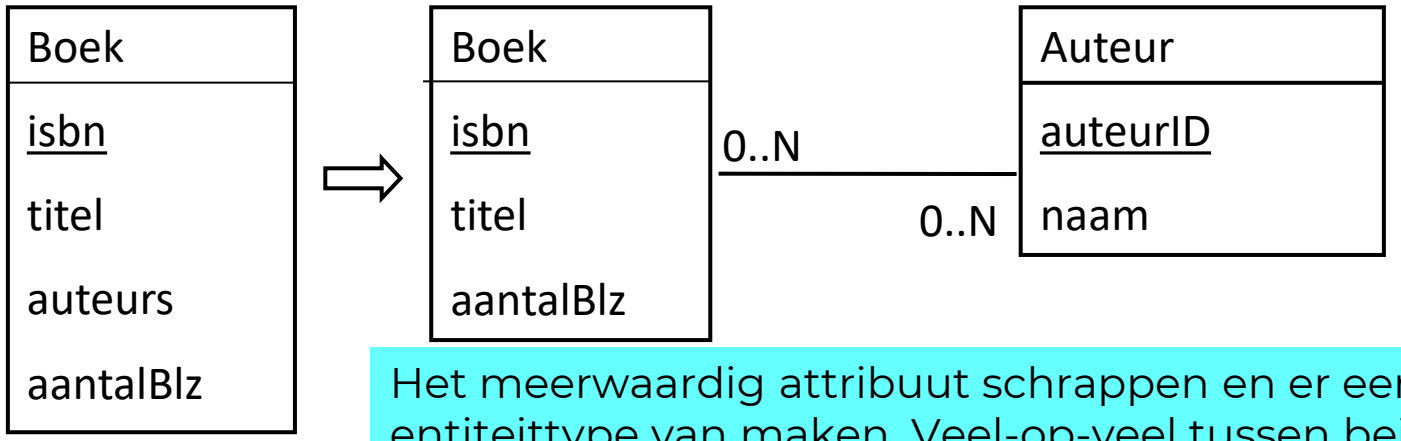
Mapping: oefening



A(<u>a1</u>, a2, c1, c2)
vs c1,c2 verwijst naar C, verplicht
B(<u>b1</u>, b2)
AB (<u>a1, b1</u>, x)
vs a1 verwijst naar A, verplicht
vs b1 verwijst naar B, verplicht
E(<u>b1, e1</u>, e2)
vs b1 verwijst naar B, verplicht
C (<u>c1, c2</u>, d1)
vs d1 verwijst naar D, verplicht, uniek
D (<u>d1</u>, d2)

HO GENT

Mapping meerwaardige attributen



Hoe?

Het meerwaardig attribuut schrappen en er een nieuw entiteittype van maken. Veel-op-veel tussen beide entiteittypes. Betekenisloze sleutel toevoegen in nieuw entiteittype. Attribuut naam of omschrijving toevoegen.

Mappingregels van veel-op-veel toepassen.

Boeken/Auteur (auteur, boek)

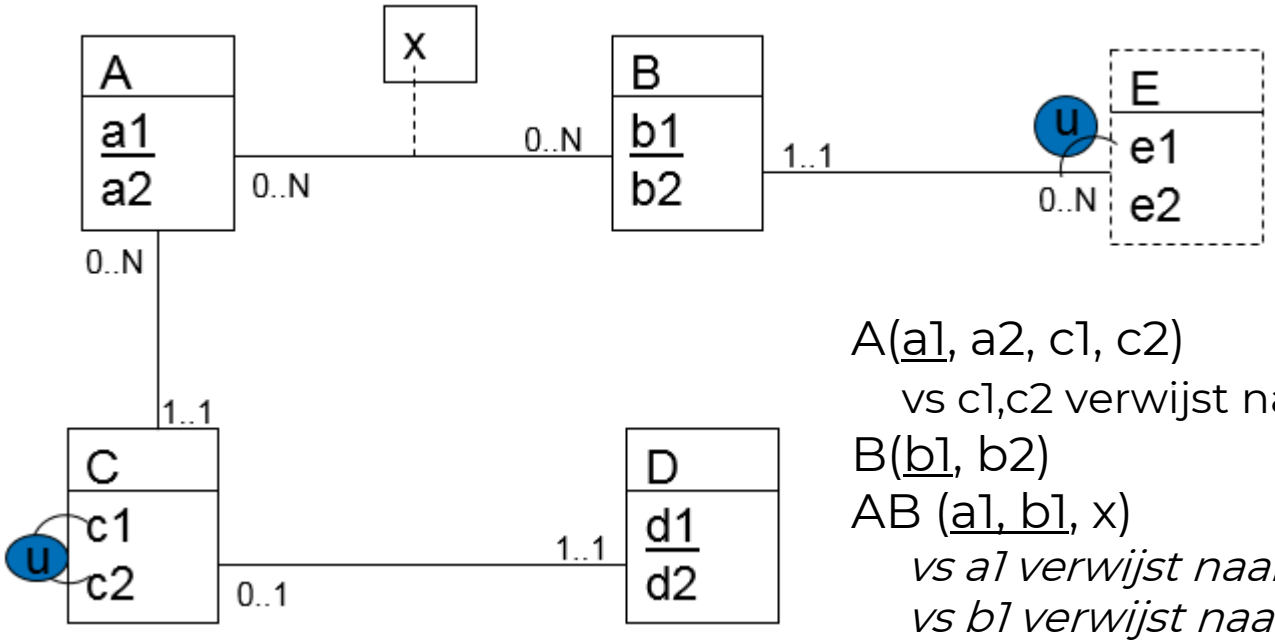
- *auteur:* vreemde sleutel, verwijst naar *auteurID* uit *Auteur, verplicht*
- *boek:* vreemde sleutel, verwijst naar *isbn* uit *Boek, verplicht*

Auteur (auteurID, voornaam, familienaam)

Boek(isbn, titel, aantalBlz)

HO GENT

Mapping: oefening



d2 is meerwaardig !

A(a1, a2, c1, c2)

vs c1,c2 verwijst naar C, verplicht

B(b1, b2)

AB (a1, b1, x)

vs a1 verwijst naar A, verplicht

vs b1 verwijst naar B, verplicht

E(b1, e1, e2)

vs b1 verwijst naar B, verplicht

C (c1, c2, d1)

vs d1 verwijst naar D, verplicht, uniek

D (d1)

D2 (ID, naam)

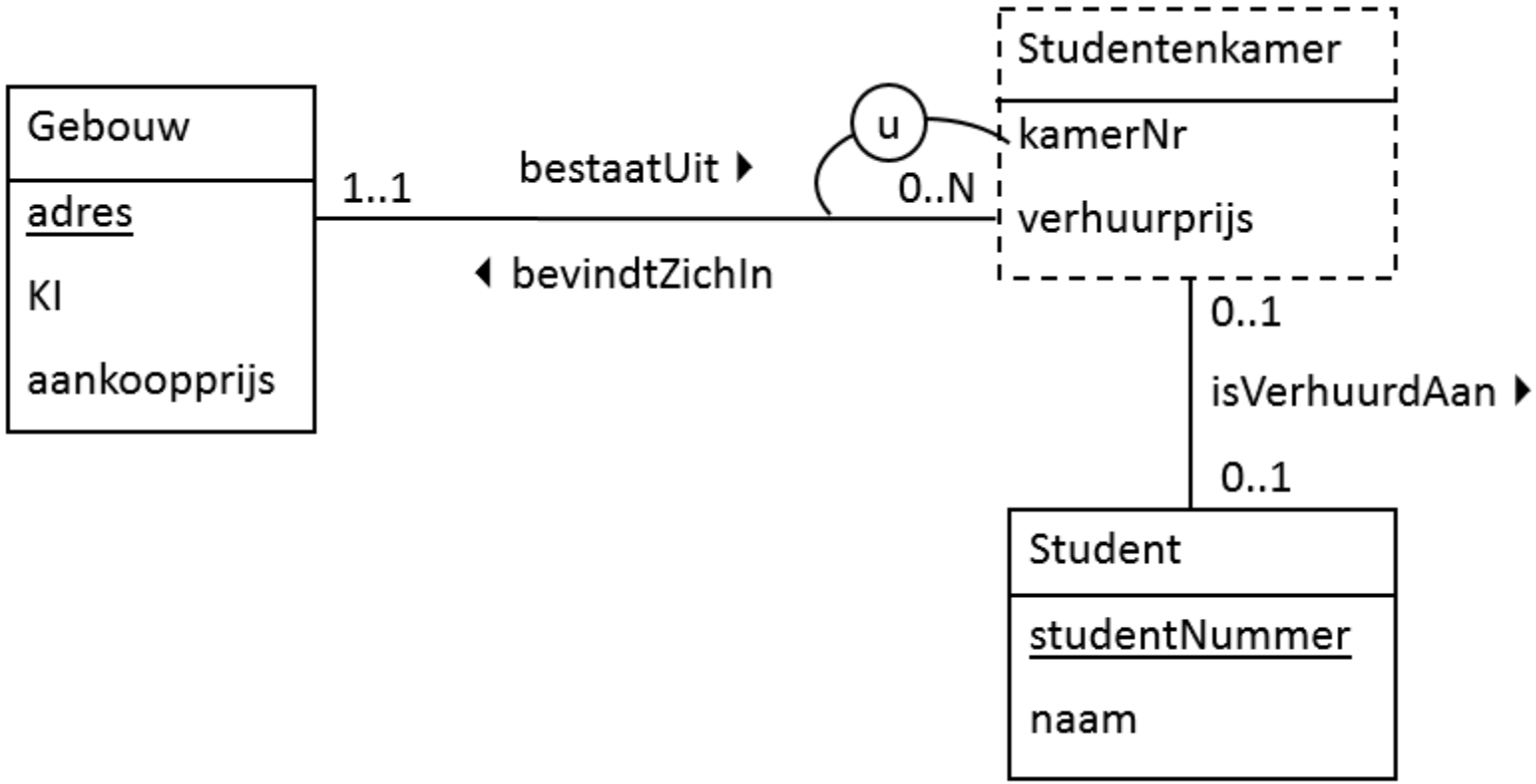
D-D2 (d1,ID)

vs d1 verwijst naar D, verplicht

vs ID verwijst naar D2, verplicht

HO GENT

Mapping: keuze van primaire sleutel



HO GENT

Mapping: keuze van primaire sleutel

Probleem …

Gebouw (straat, huisnr, postcode, gemeente, land, KI)

Studentenkamer (straat, huisnr, postcode, gemeente, land,

kamerNummer, verhuurprijs)

- straat, huisnr, postcode, gemeente, land: vreemde sleutel, verwijst naar de sleutel van *Gebouw*, is *verplicht*

Student (studentNummer, voornaam, familienaam,
straat, huisnr, postcode, gemeente, land, kamerNummer)

- straat, huisnr, postcode, gemeente, land, kamerNummer: vreemde sleutel, verwijst naar de sleutel van *Studentenkamer*, *optioneel*

HO GENT

Mapping: keuze van primaire sleutel

Andere mogelijkheid …

Gebouw (gebouwID, straat, huisnr, postcode, gemeente, land, KI)

Studentenkamer (gebouwID, kamerNummer, verhuurprijs)
gebouwID: vreemde sleutel, verwijst naar *gebouwID* uit *Gebouw,* is *verplicht*

Student (studentNummer, voornaam, familienaam, gebouwID, kamerNummer)
gebouwID, kamerNummer: vreemde sleutel, verwijst naar *gebouwID, kamerNummer* uit *Studentenkamer, optioneel*

HO GENT

Mapping: keuze van primaire sleutel

Richtlijn!

- Voorkom dat iemand anders kan beslissen over het al dan niet uniek zijn van een gekozen primaire sleutel
    - Is het ISBN van een boek werkelijk uniek? …
    - Is een barcode werkelijk uniek? …
    - Zal de barcode van een product steeds dezelfde blijven? …
- Zorg dat jij als ontwerper steeds controle hebt over de gekozen primaire sleutel.
- Bij twijfel, zelf een sleutel creëren!
    - => Surrogaatsleutel

HO GENT

Mapping van specialisatie

Participatie constraint	Disjoint constraint	Relaties
Mandatory	And	1 tabel met hierin booleans om de subtypes te onderscheiden
Optional	And	2 tabellen: 1 voor het supertype, 1 voor de subtypes met booleans om de subtypes te onderscheiden
Mandatory	Or	1 aparte tabel per subtype
Optional	Or	voor elk type een tabel

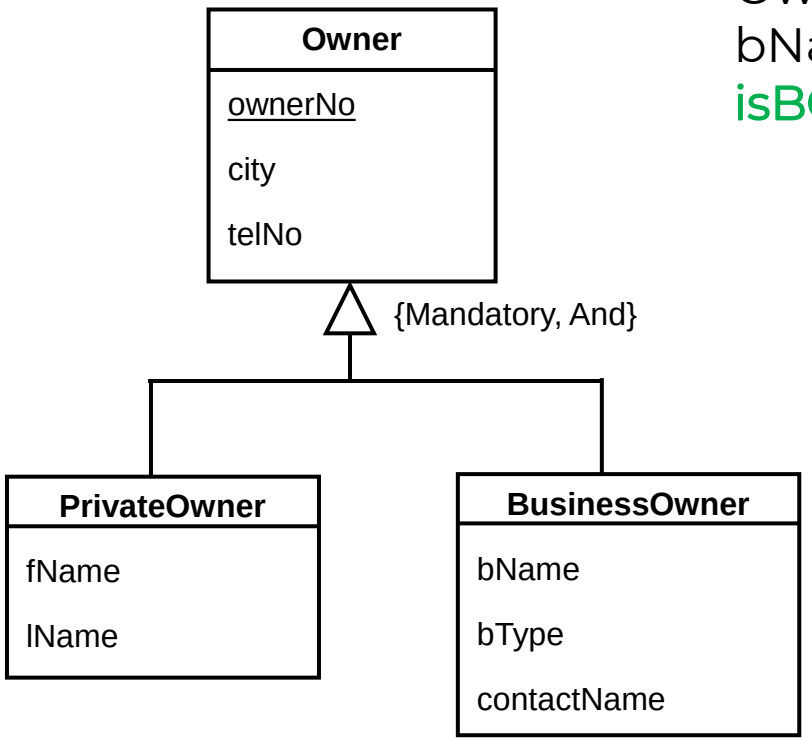
HO GENT

Mapping van specialisatie

Mandatory, And

Mandatory, And: 1 tabel met alle attributen van supertype en subtypes **+ booleans om subtypes te onderscheiden.**

Owner(ownerNo, city, telNo, fName, lName, bName, bType, contactname, isPOwner, isBOwner)



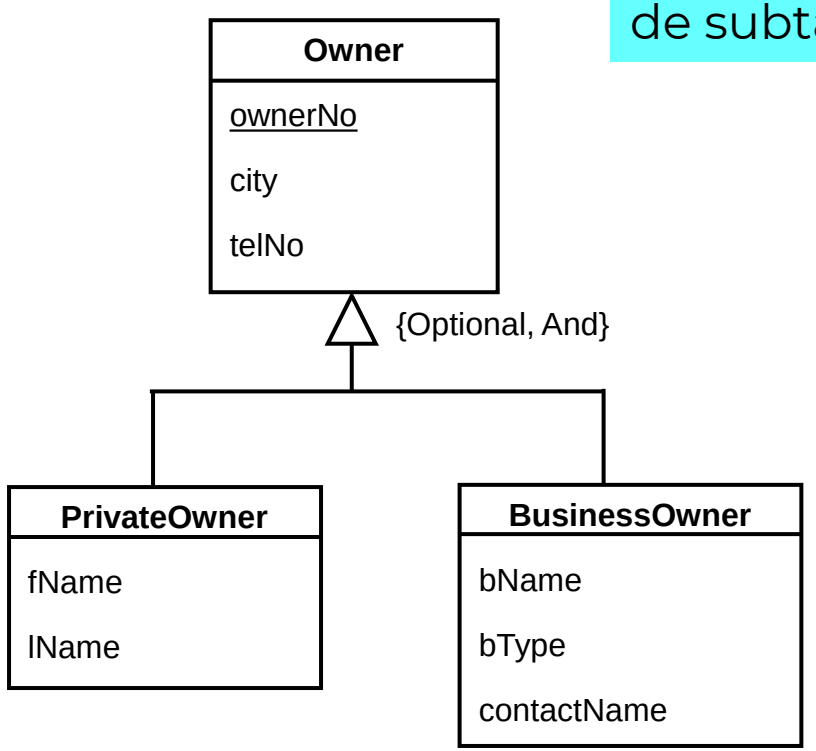
- Mandatory: ➔ bijkomende regel opnemen: 1 van beide booleans moet 'yes' zijn.
- And: Kan bekomen worden door beide booleans op 'yes' te zetten.
- Nadeel van deze oplossing: indien een owner enkel private owner is of enkel business owner➔ veel null-waarden.

HO GENT

Mapping van specialisatie

Optional, And

Optional, And: 2 tabellen: 1 voor supertype en 1 voor alle subtypes + flags om subtypes te onderscheiden. Vanuit de subtabel verwijzen naar de supertabel.



Owner(ownerNo, city, telNo)

OwnerDetails(ownerNo, fName, lName, bName, bType, contactname, isPOwner, isBOwner)

VS ownerNo verwijst naar ownerNo in owner, verplicht.

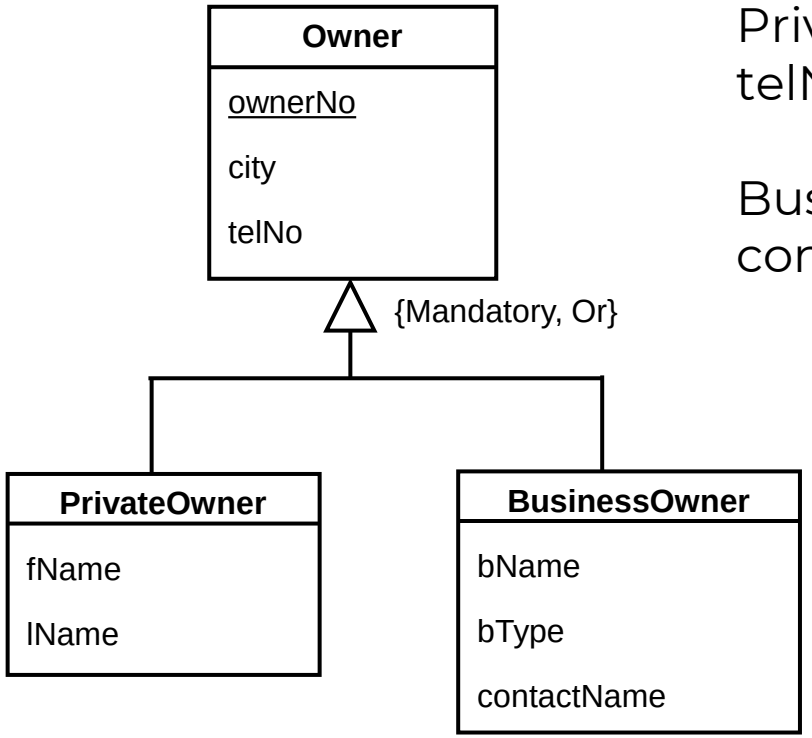
- And: Kan bekomen worden door beide booleans op ‘yes’ te zetten.
- Optional: enkel tabel Owner invullen.

HO GENT

Mapping van specialisatie

Mandatory, Or

Mandatory, Or: geen tabel voor supertype, wel een aparte tabel voor elk subtype.



PrivateOwner(<u>pOwnerNo</u>, fName, lName, city, telNo)

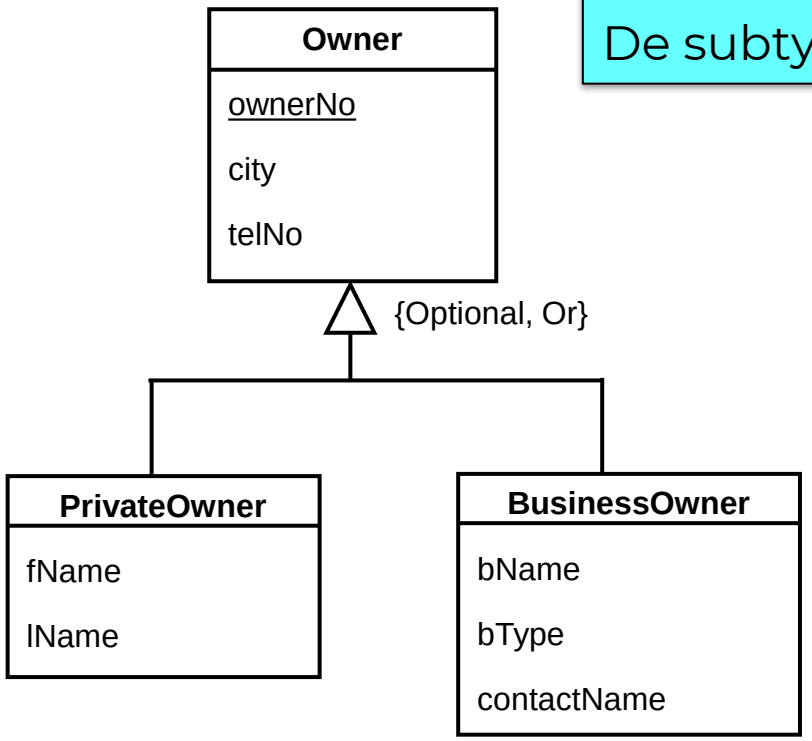
BusninessOwner(<u>bOwnerNo</u>, bName, bType, contactName, city, telNo)

- Mandatory: je kan geen andere owners dan private of business creeëren in dit model.
- Or: voor beide soorten is een aparte tabel.

HO GENT

Mapping van specialisatie

Optional, Or



Optional, Or: tabel voor elk type: zowel voor supertype als elk subtype.

De subtypes verwijzen naar het supertype.

Owner(ownerNo, city, telNo)

PrivateOwner(pOwnerNo, fName, lName)

VS pOwnerNo verwijst naar Owner.ownerNo, verplicht.

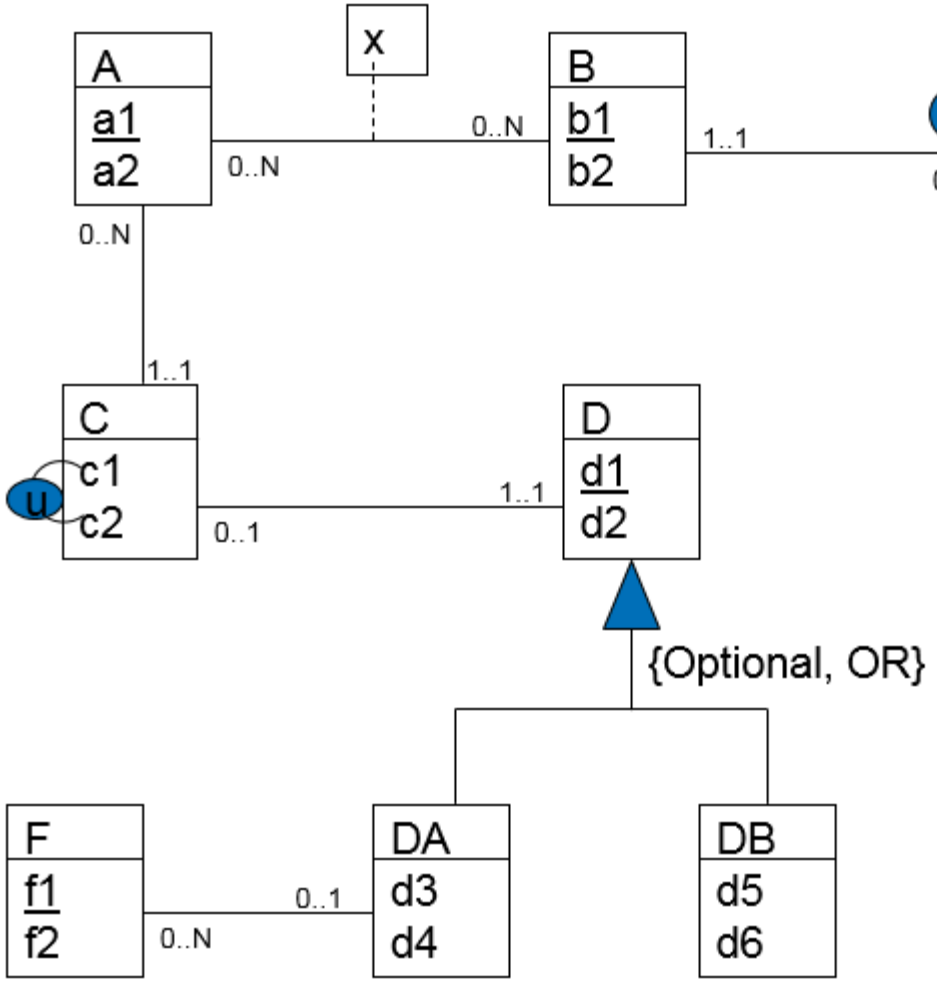
BusinessOwner(bOwnerNo, bName, bType, contactName)

VS bOwnerNo verwijst naar Owner.ownerNo, verplicht.

- Optional: Er kunnen andere owners dan private of business gecreëerd worden in de tabel Owner.
- Or: aparte tabellen voor elk subtype.

GENT

Mapping: oefening



A(a1, a2, c1, c2)

vs c1,c2 verwijst naar C, verplicht

B(b1, b2)

AB (a1, b1, x)

vs a1 verwijst naar A, verplicht

vs b1 verwijst naar B, verplicht

E(b1, e1, e2)

vs b1 verwijst naar B, verplicht

C (c1, c2, d1)

vs d1 verwijst naar D, verplicht,uniek

D (d1, d2)

DA (d1, d3, d4)

vs d1 verwijst naar D, verplicht

DB (d1, d5, d6)

vs d1 verwijst naar D, verplicht

F (f1, f2, d1)

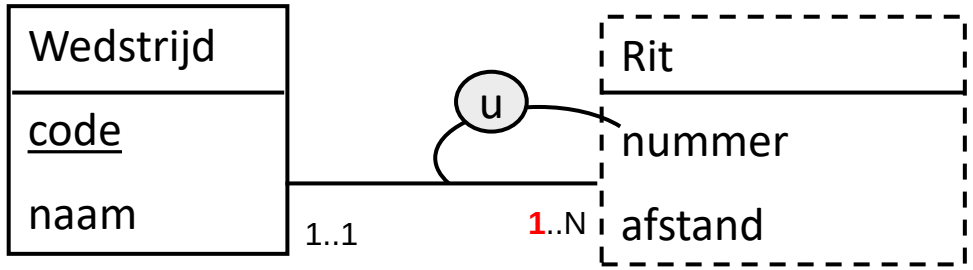
vs d1 verwijst naar DA, optioneel

HO GENT

Structuurbeperkingen

- Relationeel model bevat **enkel 0 of 1** op veel verbanden:
  - Veel op veel verbanden: opsplitsen in twee 1 op veel verbanden
  - 1 op 1 verband: beperkingsregels toepassen: ‘uniek’

- Minimum cardinaliteit 1:
  - bij max. cardinaliteit 1: beperkingsregel: mag niet null zijn
  - bij max. cardinaliteit veel: kan niet afgedwongen worden



HO GENT